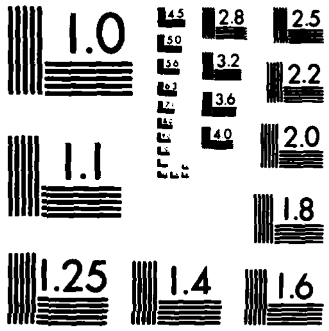

1.0
4.5
5.0
5.6
6.3
7.1
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

AFIT/GST/OS/84-10

(1)

A SIMULATION MODEL TO EVALUATE AIRCRAFT SURVIVABILITY AND TARGET DAMAGE DURING OFFENSIVE COUNTERAIR OPERATIONS

THESIS

Michael J. Foley Captain USAF
Stephen G. Gress, Jr. Captain USAF

AFIT/GST/OS/84M-10

Approved for public release; distribution unlimited

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GST/OS/84M-10	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION School of Engineering 6b. OFFICE SYMBOL *(If applicable)* AFIT/EN	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS *(City, State and ZIP Code)* Air Force Institute of Technology (AU) Wright-Patterson AFB, Ohio 45433	7b. ADDRESS *(City, State and ZIP Code)*
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL *(If applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS *(City, State and ZIP Code)*	10. SOURCE OF FUNDING NOS.
11. TITLE *(Include Security Classification)* See Box 19	

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.

12. PERSONAL AUTHOR(S)
Michael J. Foley, Capt, USAF   Stephen G. Gress, Jr, Capt, USAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(Yr., Mo., Day)*	15. PAGE COUNT
MS Thesis	FROM ______ TO ______	1984 March 16	260

16. SUPPLEMENTARY NOTATION

[illegible] public release: IAW AFR 190-17.
[illegible]   9 May 84
LYNN E. WOLAVER
Dean for [illegible] and Professional Development
Air Force Institute of Technology (ATC)
Wright-Patterson AFB OH 45433

17. COSATI CODES		
FIELD	GROUP	SUB. GR.
15	07	
15	03	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Computerized Simulation, Fighter Survivability, Airbase, Air Defenses, Airfield, Targets

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Title: A SIMULATION MODEL TO EVALUATE AIRCRAFT SURVIVABILITY AND TARGET DAMAGE DURING OFFENSIVE COUNTERAIR OPERATIONS

Thesis Chairman: James R. Coakley, Maj, USAF

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED [✓ ☒] SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER *(Include Area Code)*	22c. OFFICE SYMBOL
James R. Coakley, Maj, USAF	(513)255-3362	AFIT/ENS

DD FORM 1473, 83 APR   EDITION OF 1 JAN 73 IS OBSOLETE.   UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

Abstract

Offensive counterair missions are essential to insure air supremacy. Effective allocation of aircraft for these missions requires consideration of the likely benefits and costs. The purpose of this thesis was to develop a methodology which could be used to assess the likely target damage and the resulting attrition of friendly aircraft during offensive counterair missions. The specific problem addressed was a mission of two aircraft attacking an area target at an enemy airfield. The area of operations was contained within a ten mile radius circle centered on the airfield's runway. Located within this area were the target and ground based defenses.

A simulation model based on the SLAM language was built, but extensive use of Fortran was also made in the model. The continuous system capabilities of SLAM were used to investigate three-dimensional aircraft movements, threat engagements, and pilot reactions. The extensive use of the Fortran logic simplifies the understanding of the SLAM language required to use the model. Included in this Fortran logic is an analytical routine used to assess target damage due to weapons effects. This routine is based on a methodology contained in the Joint Munitions Effectiveness Manual.

AFIT/GST/OS/84-10

A SIMULATION MODEL TO EVALUATE AIRCRAFT SURVIVABILITY AND TARGET DAMAGE DURING OFFENSIVE COUNTERAIR OPERATIONS

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Operations Research

Michael J. Foley
Captain, USAF

Stephen G. Gress, Jr.
Captain USAF

March 1984

Preface

The purpose of this thesis was to create a methodology which could be used to assess likely target damage and friendly aircraft survival in the offensive counterair mission. We feel that the resulting SLAM simulation model effectively combines weapons effects logic of the Joint Munitions Effectiveness Manual with a credible method of assessing the survivability of aircraft that are exposed to the ground based threats around enemy airfields.

We would like to express our gratitude for the patience and support from our wives and children during the many evenings and weekends which were required to complete this thesis. We also appreciate the interest and support of our advisor, Maj. James R. Coakley of the Air Force Institute of Technology. Finally, we wish to thank Professor D. Walter Breuer, also of the Air Force Institute of Technology, for his selfless assistance and sound advise.

Michael J. Foley
Stephen G. Gress, Jr.

Table of Contents

List of Figures

List of Tables

Abstract

Offensive counterair missions are essential to insure air supremacy. Effective allocation of aircraft for these missions requires consideration of the likely benefits and costs. The purpose of this thesis was to develop a methodology which could be used to assess the likely target damage and the resulting attrition of friendly aircraft during offensive counterair missions. The specific problem addressed was a mission of two aircraft attacking an area target at an enemy airfield. The area of operations was contained within a ten mile radius circle centered on the airfield's runway. Located within this area were the target and ground based defenses.

A simulation model based on the SLAM language was built, but extensive use of Fortran was also made in the model. The continuous system capabilities of SLAM were used to investigate three-dimensional aircraft movements, threat engagements, and pilot reactions. The extensive use of the Fortran logic simplifies the understanding of the SLAM language required to use the model. Included in this Fortran logic is an analytical routine used to assess target damage due to weapons effects. This routine is based on a methodology contained in the Joint Munitions Effectiveness Manual.

I Introduction

Background

In the basic doctrine of the United States Air Force, it is noted that offensive counterair operations are essential in order to attain air supremacy (Ref 2:Ch 2,15). These counterair operations are air operations which attempt to destroy the enemy air force's aircraft and support activities. The typical targets for such operations are the aircraft, equipment, facilities and supplies which are located at enemy airfields. The accurate determination of such targets and the suitable use of aircraft and weapons against them is a critical function which requires complex decisions (Ref 6:v).

Such decisions have a natural impact on other combat operations. The commitment of sorties to the offensive counterair operations results in fewer aircraft available to defend friendly airspace or to support friendly ground forces. Therefore, the Air Force planner who allocates sorties must be aware of the likely benefits as well as the likely costs associated with commitment of the sortie resources against specific offensive counterair targets. His goal must be "to achieve the best possible tradeoff between results and costs" (Ref 6:8).

On 21 April 1983 Brigadier General W. L. Goodson, then Deputy Chief of Staff for Plans, United States Air Forces in Europe (USAFE) reiterated the requirement for an accurate estimate of sortie effectiveness and losses in such missions as offensive counterair. Such estimates result in a better understanding of how critical missions compete for the limited aircraft resources (Ref 9).

In making these sortie allocation estimates, the planner must consider the tradeoff between the probability of destroying the target as well as the likely damage or attrition suffered by friendly aircraft in such missions. The primary tool currently used by Air Force planners in assessing likely target damage is the Joint Munitions Effectiveness Manual (JMEM) (Ref 1). The manual provides an estimate of target damage as a function of various weapons and aircraft release parameters. However, the same manual fails to provide another major aspect in assessing the tradeoff between the benefits and costs of allocating sorties to offensive counterair missions. That is, the manual does not address the probability of an aircraft actually arriving at the release point and subsequent survival during exit from the target area.

Problem Statement

A planning tool must be found or developed which demonstrates the interaction between the major factors of target destruction and friendly aircraft attrition in the offensive counterair role. Such a tool which addresses both the likelihood of target damage as well as the ability of that aircraft to arrive at the weapons release point and then to exit the target area can provide valuable information to the Air Force planner. He can use this information to understand the tradeoffs between target damage and aircraft attrition as he attempts to allocate limited aircraft resources.

Objective

The primary purpose of this research is to develop a methodology which demonstrates the likely target damage and friendly aircraft attrition in the offensive counterair role. As the conventional weapon damage logic from the Joint Munitions Effectiveness Manual (JMEM) is the most credible source of target damage in use by the Air Force today (Ref 1), an attempt will be made to incorporate this logic into a realistic methodology. The scenario for this methodology will reflect the aircraft's susceptibility to the likely threat systems which exist in the area around enemy airfields. Further, since given a specific target and threat scenario the resulting target damage and aircraft attrition will depend in large part on the type of attack profile (tactic) and weapons used, the research will address various combinations of weapons and tactics.

Literature Review

Existing Methodologies. As noted earlier, JMEM is a widely accepted methodology for assessing levels of target destruction given various types of targets, aircraft, weapons and release conditions. Since JMEM does not model susceptibility of aircraft to ground based threats, the logic of JMEM alone is insufficient for this research.

The TAC Repeller Model is an attrition model which reflects capabilities of surface-to-air missiles (SAM) systems and anti-aircraft artillery (AAA) against aircraft. Detection and allocation of aircraft on prespecified flight paths is accomplished in the model (Ref 18:711-712). In does not appear that this model incorporates aircraft defensive maneuvers in the assessment of attrition. Further, as this model does not incorporate damage assessment of the ground target, the model alone is not appropriate for this research.

TAGSEM is a model developed by Air Force Systems Command to determine target damage and weapons effectiveness of various air-to-ground systems. This model requires inputs of both weapon lethalities and aircraft survivability against various threat systems (Ref 18:735-736). Since this model needs inputs which are in fact among the desired outputs of this research, TAGSEM is not appropriate.

TAC Warrior is a large campaign model which addresses all aspects of air warfare. Among these aspects is the offensive counterair mission. The inputs required for this model are massive in number. The model appears designed for research of theater level warfare. Its use, therefore, to identify the type of relationships required for this research would be cumbersome and inefficient.

The Airbase Damage Assessment Model (AIDA) was developed by the Rand Corporation in 1976 to assess target damage due to conventional munitions. The model may be operated in either Monte Carlo or deterministic modes of operation. Neither aircraft survivabiliy nor probability of seeing the target are assessed in the model. Calculations are based on assumptions which the Rand Corporation claims are equivalent in accuracy to the hand calculation method of JMEM (Ref 7:2). As both AIDA and JMEM address the same basic question of conventional bomb damage, this research will incorporate the JMEM methodology due to the credibility and wide use of JMEM in Air Force squadrons which are tasked for offensive counter air operations.

Previous Theses. In 1981 Leek and Schmitt investigated the employment of a Forward Looking Infrared (FLIR) equipped aircraft on a night battlefield interdiction mission. Their work addressed the aircraft survivability when exposed to a typical array of enemy defenses in the forward edge of the battle area (FEBA). They incorporated a detailed SAM and AAA scenario. This scenario included a detailed approach to calculate radar cross section as a function of aircraft profile with respect to the SAM or AAA site. Their emphasis was on the survivability of the aircraft when exposed to the FEBA defenses. Their work did not address actual target damage inflicted by the aircraft.

In 1982 Anderson and Nenner extended the Leek and Schmitt work by incorporating Wild Weasel aircraft to provide threat suppression for a force of strike aircraft. The threat scenario was similar to that of the 1981 thesis but involved a more sophisticated command and control decision process for the launch of threat missiles against aircraft. The measure of merit was the number of attacking aircraft that reached the target area as a result of Wild Weasel aircraft suppression of the enemy threats. Aircraft damage to the target was not assessed.

In 1983 Neal and Kizer produced a thesis in which they addressed the tradeoff between aircraft survivability and target damage in the close air support (CAS) mission. The aircraft were engaged by FEBA ground based threat systems similar to those described in the 1981 and 1982 theses. Neal and Kizer introduced the concept of a three-dimensional attack by the aircraft against the CAS target. They also simplified the logic required to fire missiles at the aircraft. In addition, these authors analyzed the actual damage of point targets inflicted by the aircraft. These point targets were individual tanks. Weapons used against the tanks were Maverick missiles and cannon.

This current research of the offensive counterair mission will incorporate some of the concepts developed in these theses from 1981, 1982, and 1983 as well as introduce improvements and concepts not treated before. Similar to the work by Neal and Kizer, this effort will use a three-dimensional model to investigate target damage versus aircraft attrition. In addition, the capabilities of the enemy missiles will be similar to those described in the previous theses.

Unlike the earlier theses, however, the effort in this research will move away from the FEBA and concentrate on the events in the immediate area of the target airfield. The release of strings of bombs will be introduced and the assessment of the resulting damage made via logic from JMEM. Given a specific target and threat environment, this research will address the flight planning requirements for a coordinated attack of two aircraft flying either similar or different tactics against an area target on the airfield. An area target is a grouping of similar targets as opposed to point targets which are individual targets. The research will introduce the capability for the pilot to decide whether or not to evade a threat missile and then to execute a defensive maneuver if desired. A method to assess probability of SAM kill against a maneuvering aircraft will be introduced to augment the method of assessing the probability of kill against a nonmaneuvering aircraft introduced in earlier theses. Unlike previous theses, this research will consider the launch and guidance of more than one missile at a time from SAM site against an aircraft. Finally, this research will seek to improve the three dimensional relationship between the aircraft and missile used in computing the predicted impact point of the missile and the aircraft.

Scope

This research will address a flight of two aircraft assigned to destroy an area target at an enemy airfield. The location of the target, the dimensions of the target, and the types and locations of the threats will remain fixed as defined in the scenario. The research will then investigate the level of target destruction inflicted by the two aircraft. The aircraft attrition for the same mission will also be investigated.

This analysis will only deal with the events in the immediate target area. This target area is a ten nautical mile radius circle centered on the center of the runway. No activity outside this target area will be considered. Rather, the capability of these two aircraft to survive to the weapons release point, to inflict damage on the target, and to safely exit the ten mile radius will be assessed.

The probabilitity of target damage, will, therefore, incorporate both the probability of the aircraft arriving at the target as well as the likely weapons effects against the target. In addition, the following concepts will apply:

1. Each aircraft is a single seat, clear air mass fighter with aircraft performance, avionics and bombing systems similar to the F-16 aircraft.

2. Two possible weapons loads will be investigated: eight 500 pound low drag bombs and eight 500 pound high drag weapons.

3. For any one mission, both the lead (aircraft #1) and

wingman (aircraft #2) will carry the same weapons load.

4. Three possible attack options will be investigated. Tactic 1 will be a level delivery for the lead and a level delivery for the wingman. Tactic 2 will be a level delivery for the lead and a pop-to-angular delivery for the wingman. Tactic 3 will be a pop-to-angular delivery for lead followed by the same delivery for the wingman.

5. The dive angle for the angular delivery with high drag weapons will be 10 degrees, while the dive angle for the low drag angular delivery will be 15 degrees.

6. The assessment of damage to the target area will be based on calculations contained in the Joint Munitions Effectiveness Manual.

7. The effects of target area weather will not be modeled. The weather will be good enough for level, 10 degree, and 15 degree angular deliveries to be performed.

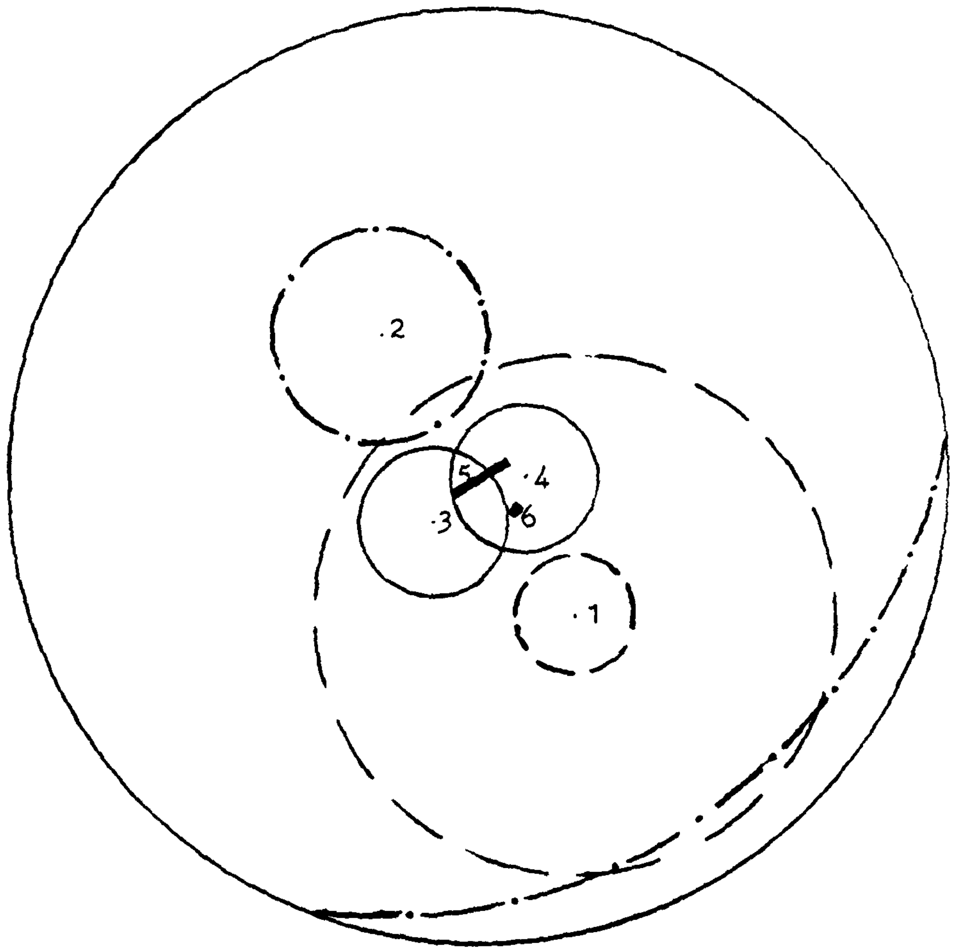
8. Only radar controlled SAM and AAA threat systems will be included on the analysis. These will be fixed sites in defense of the enemy airfield.

Scenario

Two different aircraft will be assigned to attack an area of petroleum, oil, and lubricant (POL) storage tanks. The length of this area of tanks is 550 feet and its width is 400 feet. The center of the POL area is located 4000 feet east and 4000 feet south of the runway center. This runway center is designated as the center of the 10 nautical mile radius target area. The length of the POL tanks is oriented 045-225 degrees. SAM site 1 is positioned 12,000 feet east and 18,000 feet south of the center of the runway. SAM site 2 is located 12,000 feet west and 18,000 feet north of the runway center. The first of two AAA sites will be located 6000 feet east and 0 feet north of the runway. The second AAA site will be located at 6000 feet west and 6000 feet south of the runway. A depiction of the target area elements is contained in Figure 1.

Each pilot will enter the area at an assigned time, fly a flight planned route to the target, deliver his weapons from his preplanned delivery tactic, and then depart the target area as expeditiously as possible. At any time in the target area either or both aircraft may be engaged by the SAM and AAA threats. If engaged, each pilot will determine if evasive action is required and take such action if the threat warrants it.

A probability of target damage will be determined for each aircraft and a probability of aircraft kill will be assessed against each aircraft. These probabilities for each aircraft will be combined to form the probability of target damage for the mission of two aircraft as well as the probability of aircraft survival for the mission.



1 = SAM Site Number One
2 = SAM Site Number Two
3 = AAA Site
4 = AAA Site
5 = Runway
6 = Target
----- = Maximum and Minimum Ranges of SAM Site Number One
-.-.- = Maximum and Minimum Ranges of SAM Site Number Two
—— = Maximum Ranges of AAA Sites

Fig. 1. Depiction of Target Area

Summary

Offensive counterair operations are essential in order to attain air supremacy. The allocation of aircraft resources for the offensive counterair mission detract from the number of aircraft available to perform other critical missions. In allocating aircraft to the offensive counterair mission, the planner needs a planning tool which calculates the tradeoff between probable target damage versus the ability of aircraft to survive that mission. The goal of this research is to develop a planning tool which can be used to calculate both the probability of target damage due to weapons effects as well as the probability of aircraft survival to the weapons release point and then out of the threat area.

II System Structure

An analysis of the target damage and aircraft attrition which occurs during a mission of two aircraft must address various major topics. These topics can be divided into four major categories: attack planning, assessment of target damage per aircraft, probability of each aircraft surviving, and assessment of overall target damage and aircraft attrition for the mission.

Attack Planning

Target, Weather, Terrain, and Tactics. The first step must be the identification of the target. In addition to the target's location, this must include some estimate of the target's dimension and physical characteristics. This information can be used to decide what type of weapons, what type of fuzing on those weapons, and what type of weapons delivery would be most appropriate to maximize target damage.

Terrain in the target area may require that delivery tactics be modified. Rugged terrain may prohibit certain types of attacks or cause problems in the visual acquisition of the target. However, since the terrain around an enemy airfield should be relatively flat these adverse effects of terrain should be minimal.

In Europe, the weather is usually a significant factor. Tactics which optimize target damage may require modification to allow visual deliveries during periods of low ceilings or reduced visibility.

Threats can also dictate that one tactic be favored over another. Typical threat systems defending an airfield incorporate some combination of antiaircraft artillery (AAA) and surface-to-air missile (SAM). As with weather, the type and level of enemy threats may require that a tactic be varied from that which promises the optimum target damage.

Axis of Attack. The axis of attack should optimize the anticipated damage to the target. However, threats, and to a lesser extent, terrain and weather, may dictate that certain axes of attack be used or eliminated from consideration. In addition, for a flight of two aircraft, it is desirable that both aircraft not fly the exact same ground track to attack the target. Similar ground tracks usually increase the ease with which enemy threat systems can acquire and engage the second aircraft. Another consideration is the direction of an aircraft's turn onto its final attack heading. If possible, particularly during a climbing, or "pop-up" maneuver, planning should minimize the amount of time which the aircraft is "belly-up" to the primary threats. That is, since the main concentration of threats, espcially AAA, will be near the runway, the planning should minimize the time during which the pilot will lose visual contact with that runway. Finally, given different attack headings for the first and second aircraft, the attack headings should allow the wingman to look in the direction of the lead aircraft and still have the runway within his field of view.

Type Weapons. A wide variety of types of targets exist in the airfield environment. These vary from hardened structures such as concrete aircraft shelters to soft targets such as unsheltered aircraft. The runway and taxiway surfaces are also possible targets. As varied are the targets, so also are the weapons which can be used against them. The most common weapons still used are the relatively inexpensive general purpose bombs. The typical types used are the 2000 pound MK-84 and the 500 pound MK-82. Both a low drag and high drag option are available on the MK-82. More sophisticated versions of these general purpose bombs are the highly accurate but expensive laser guided bombs. Where the MK-82 and MK-84 bombs are used against most any target on the airfield, laser guided bombs are typically used against hardened, high value targets such as command and control facilities. Common weapons used against soft targets are cluster bomb units (CBU) such as CBU-58. CBU are small bomblets housed in a large canister. The canister opens and dispenses the bomblets prior to ground impact. The bomblets may then detonate on ground impact or may incorporate a timing delay prior to detonation. These weapons, in addition to the aircraft's 20mm or 30mm cannon, are the most common conventional weapons in the current Air Force inventory for use against airfield targets.

Type of Delivery. In the determination of the type of delivery, the desired target damage given a specific type of weapon is an important factor. However, as in determining the axis of attack, the weather and threats may dictate a delivery that is less than optimum for the desired target damage. Prior to the Vietnam War, dive bomb deliveries of 30 and 45 degrees were common. These patterns require roll-in altitude of 7000 to 10,000 feet above ground level (AGL) and release altitudes of 3000 to 5000 feet AGL. Low ceilings in Vietnam often precluded use of these altitude regimes for visual bombing. As a result, more emphasis was put on low angle bombing of 5 to 20 degrees of dive angle (Ref 15:26). In addition to the problem with weather, 30 and 45 degree dive bomb patterns were more vulnerable to SAM and AAA damage. These patterns did, however, offer a higher degree of survivability against small arms fire than do level and low angle deliveries (Ref 15:26).

The inflight profiles, therefore, for the level and low angle deliveries are designed to minimize susceptibility to SAM and AAA threats, remain below low ceilings, and yet result in satisfactory target damage.

Level Delivery. The low altitude level delivery is designed to minimize exposure to SAM and AAA systems and remain below low ceiling. As depicted in Figure 2, the delivery begins with a low level ingress at minimum altitudes (200 feet and below) to delay detection by enemy radar systems and minimize the number of SAM systems which can guide weapons at those altitudes. At some point prior to the target however, a climb is required to attain the desired altitude for the release of the weapons. The minimum weapons

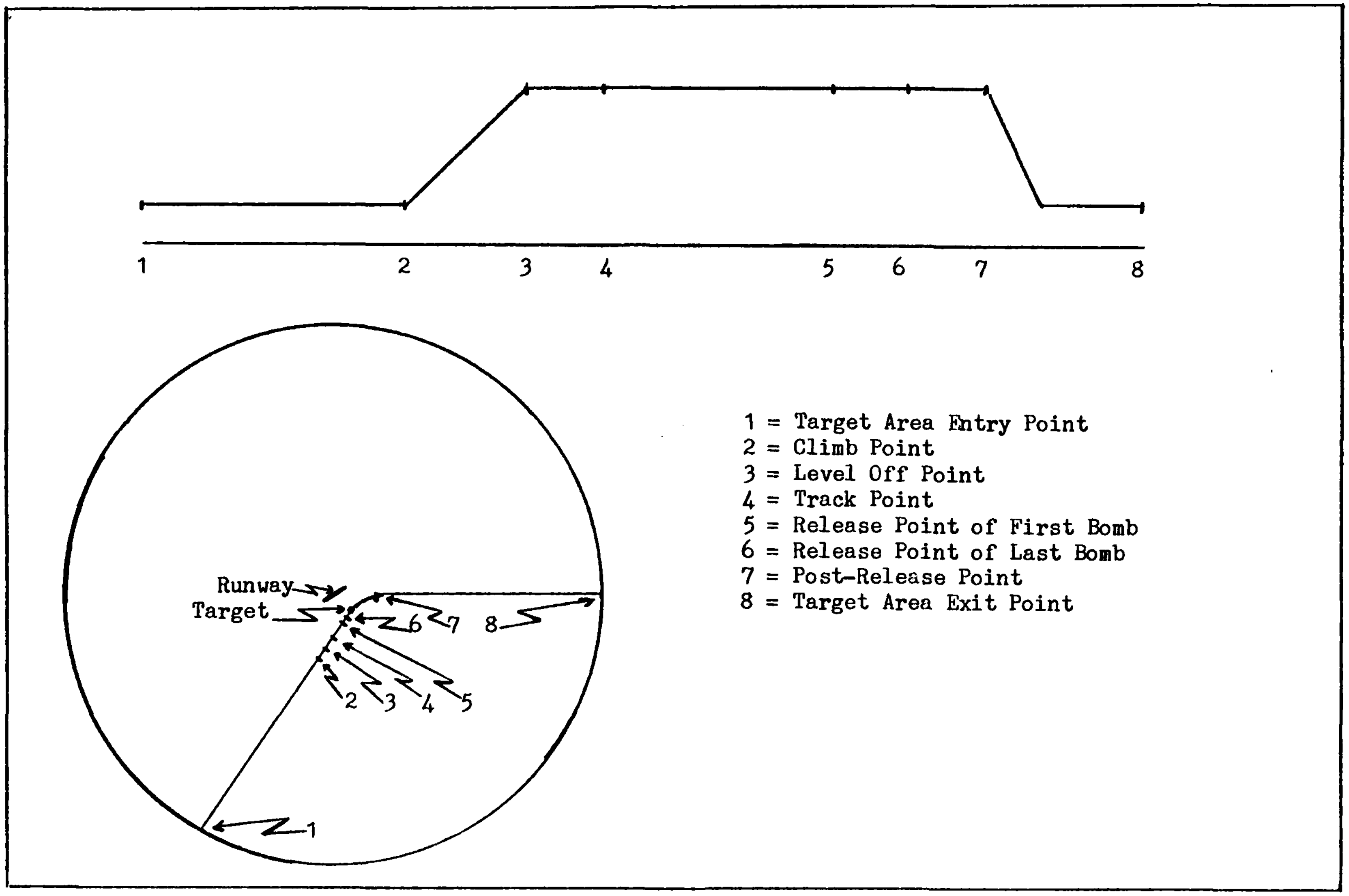

1
2
3
4
5
6
7
8
Runway
Target
1 = Target Area Entry Point
2 = Climb Point
3 = Level Off Point
4 = Track Point
5 = Release Point of First Bomb
6 = Release Point of Last Bomb
7 = Post-Release Point
8 = Target Area Exit Point

Fig. 2. Profile of Level Attack

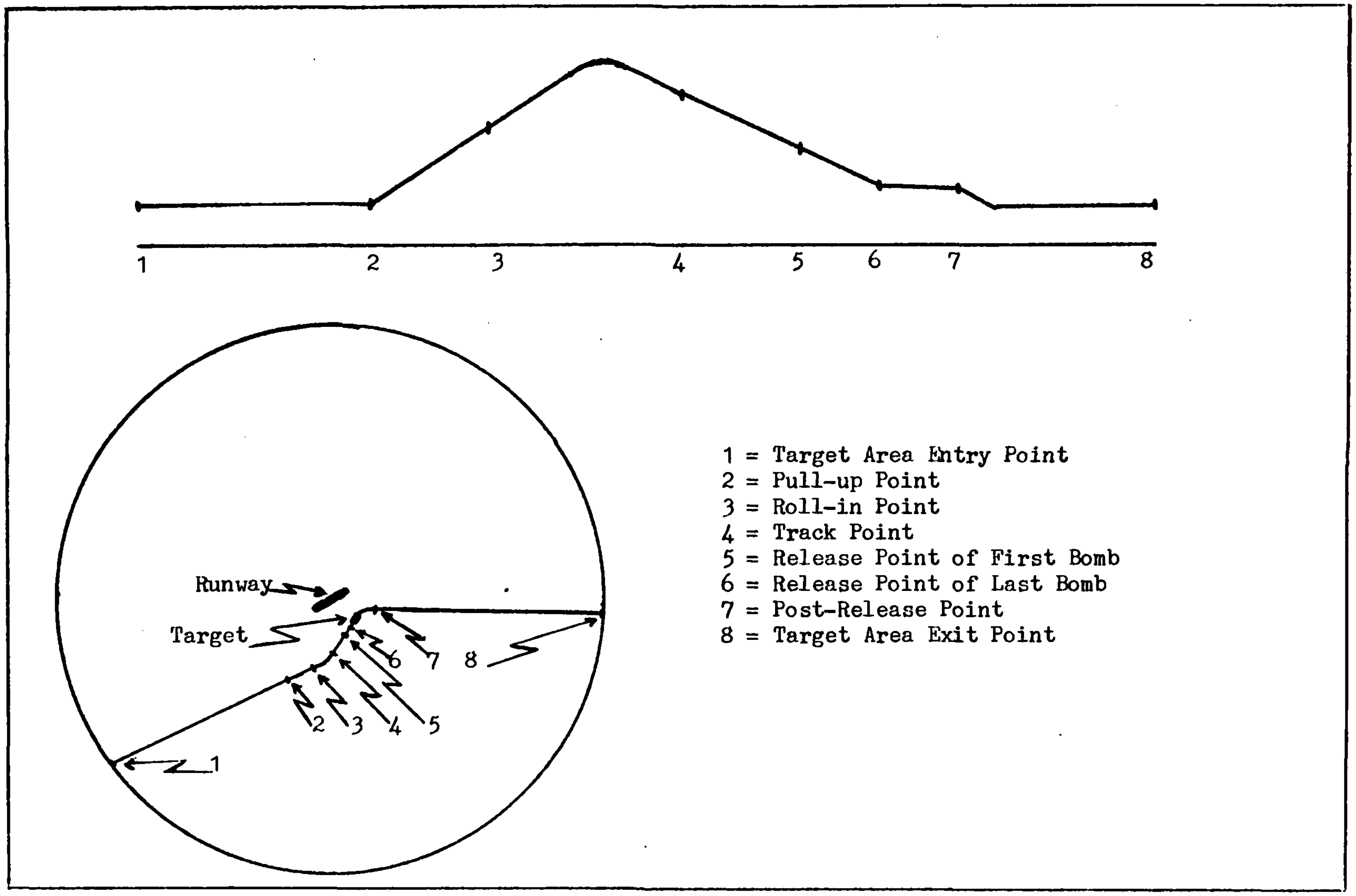
release altitude is a combination of the minimum altitude required for the released bombs to arm and for safe escape of the aircraft from the resulting weapon fragmentation pattern (Ref 15:29). This altitude will vary with weapon type. For instance, high drag weapons have lower minimum release altitude than do low drags. It is desired that the aircraft level-off and arrive at the track point at about five seconds prior to release of the first weapon. It is at the track point that the pilot attempts to track his aiming pipper toward the target so that the pipper is on the target when he initiates the release of the weapons. During the tracking time the pilot attempts to maintain his parameters of altitude, dive angle, and airspeed so as to obtain the desired weapon affects on the target. After the release of the weapon the pilot executes a level 4 to 5 G hard turn away from the target. This is followed by a return to the original altitude of below 200 feet for exit from the target area.

Although the level delivery results in reduced vulnerability to most radar aimed threat systems, the delivery also results in delayed visual acquisition of the target by the pilot. As noted in USAF Fighter Weapons School texts, visual acquisition of the target during a high speed attack gets progressively more difficult as the altitude of the aircraft is reduced (Ref 23:Ch 3,3). This is because the pilot on a level attack must find the target on the horizon, whereas the pilot on a diving attack can look down onto the target. Since the pilot must locate the target on the horizon in a level attack, a target which lacks vertical development or contrast may not be seen in time for the pilot to deliver the weapons.

Low Angle Delivery. Since the low angle delivery is a diving delivery, it offers increased target acquisition compared to the level attack. The low angle attack also yields greater delivery accuracy than that produced by level deliveries (Ref 23:Ch 3,2). The low angle attack does, however, result in increased altitude as compared to level deliveries. This increased altitude results in increased exposure to SAM and AAA systems.

In an attempt to minimize the increased exposure to threats, a low angle delivery is typically entered via a pop-up attack. The approach to the pop-up is typically via a low altitude ingress similar to that of the level attack. However, the ingress heading for the pop pattern is typically offset 15 to 30 degrees from the desired final attack heading. The typical airspeed is 500 to 550 knots.

On a typical pop-to-angular attack (Fig. 3), the ingress heading and altitude are maintained until the pull-up point at which time the pilot initiates a 3 to 5 G pull to the desired climb angle (Ref 11:Ch 6,4). This climb angle for a 15 degree or less dive angle should equal the dive angle plus five degrees (Ref 23:Ch 5,10).



1 = Target Area Entry Point
2 = Pull-up Point
3 = Roll-in Point
4 = Track Point
5 = Release Point of First Bomb
6 = Release Point of Last Bomb
7 = Post-Release Point
8 = Target Area Exit Point

Fig. 3. Profile of Pop-to-Angular Attack

$$\text{CLIMB ANGLE} = \text{DIVE ANGLE} + 5 \tag{1}$$

where CLIMB ANGLE and DIVE ANGLE are in degrees.

The pilot continues to climb until he reaches a roll-in altitude. At the point that the pilot passes this altitude, he executes a roll toward the target and begins to pull back down as he attempts to change his pitch to his desired dive angle. Approximately halfway through his turn to the target the aircraft will apex in altitude. The apex altitude is used to determine the roll-in altitude. Both of these altitudes can be predicted (Ref 23:Ch 5,10-11). For a dive angle of 15 degrees or more, the apex altitude is found by the equation:

$$\text{APEX ALT} = 2 \times \text{DIVE ANGLE} \times 100 + (\text{RELEASE ALTITUDE}/2) \tag{2}$$

For a 10 degree dive angle a more appropriate equation is

$$\text{APEX ALTITUDE} = \text{RELEASE ALTITUDE} + 1000 \text{ FEET} \tag{3}$$

In either case, the resulting roll-in altitude is

$$\text{ROLL-IN ALTITUDE} = \text{APEX ALTITUDE} - (60 \times \text{CLIMB ANGLE}) \tag{4}$$

As the pilot rolls out he arrives at the track point. As with the level delivery, this should allow about five seconds to track his pipper toward the target and correct his release parameters to obtain desired weapons effects. After the last weapon is released the pilot pulls the nose of the aircraft up towards the horizon as he executes a 4 to 5 G turn away from the target. He then descends to 200 feet or below to exit the target area.

This pop-to-angular profile (Fig. 4) results in greater exposure of the aircraft to AAA and SAM systems, than does the low altitude level delivery, but increases the likelihood of visually acquiring the target. It also results in improved bomb impact angles (Ref 15:31).

Release Parameters. Regardless of whether the pilot is performing a level or a pop-to-low angle attack, he will plan to release the weapons at predetermined release conditions. The primary parameters are the dive angle, release airspeed, and altitude. In addition, the pilot will program the aircraft avionics systems and bomb delivery computers to release the weapons with the number of release pulses to be generated to the bomb racks and the number of bombs to be released per release pulse. The intervolometor setting, the times between the release pulses, will result in a preplanned distance between bomb impact point if the pilot meets his preplanned parameters.

Route of Flight. Once the pilot has determined his targets location, decided on an attack axis and delivery tactic as a function of desired target damage, weather, and threats, he will flight plan back to some significant initial point (IP) close to the target. This point is usually a

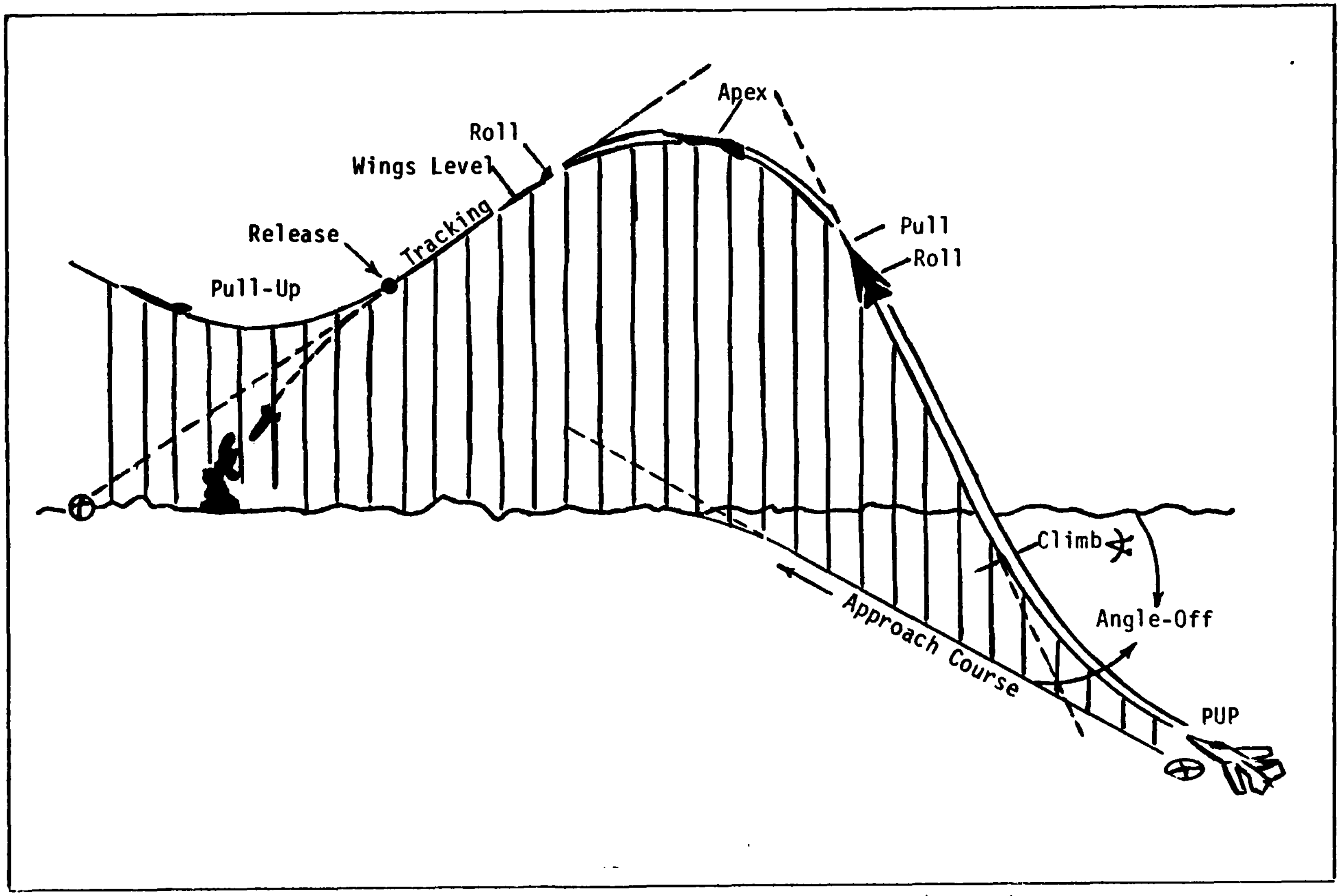


Fig. 4. Pop-to-Angular Delivery (Ref 23)

visually significant point which the pilot uses to update his position. He will usually select a significant point, or at least a reference heading to turn to as he departs the target. He will then plan the route of flight to insure he arrives at the IP and has a place to go after he leaves the target area.

In the case of a multiple ship attack, deconfliction must also be addressed. Deconfliction is the scheduling of aircraft arrival times at the target such that pilots of follow-on aircraft do not fly over the target while there is danger of fragmentation from the previous aircraft's bombs. Figure 5 exhibits a typical fragmentation pattern. The depiction includes the times, heights, and ranges from the target that bomb fragmentation could pose a danger to a follow-on aircraft.

Probability of Target Destruction

The factors contributing to target destruction are summarized in an analytical method used in the Joint Munitions Effectiveness Model (JMEM) (Ref 20:Ch 4,64-65). The single sortie probability of destruction is a function of both the probabilitity of damage given the fact that the airplane arrives at the release point ready to release weapons, as well as the probability that the aircraft actually arrives at the release point.

The four primary elements in the probability of target damage are delivery reliabilitity, conditional damage

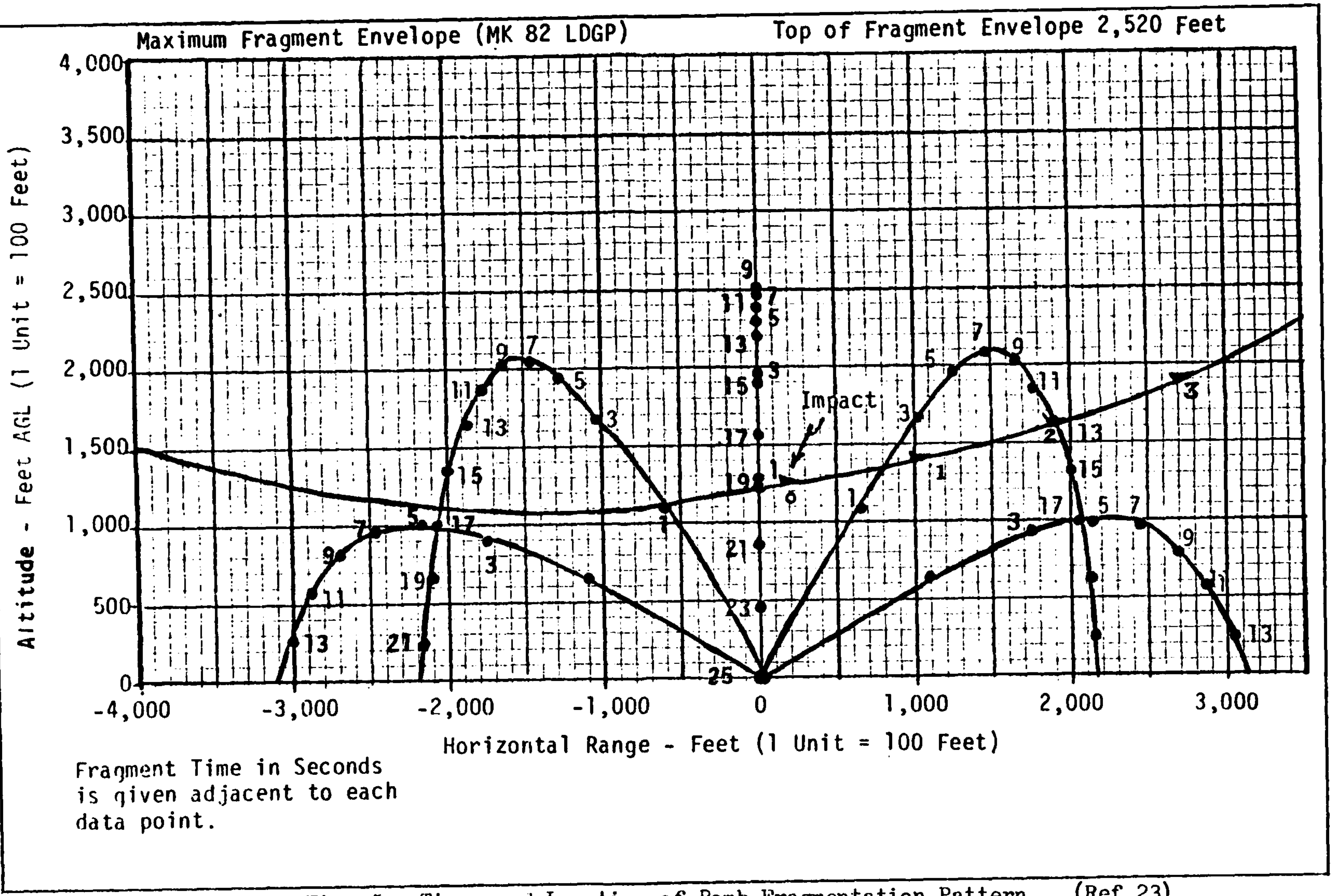


Fig. 5. Times and Location of Bomb Fragmentation Pattern (Ref 23)

probability, effective weapons pattern, and fractional target coverage. The delivery reliabilitiy is the percentage of sorties whose delivery accuracy is characteristic of the given circular error probable (CEP). This CEP is the radius of a circle whose center is on the target and contains 50 percent of the bombs that are dropped to impact directly on the target. The conditional damage probability is the probability that target damage does occur within the pattern of bombs dropped in the stick or string. The effective weapon pattern is a value greater than or equal to the width and length required to be effective against the target. For example, if the target width is less than the bomb pattern width, then the effective weapon pattern width equals the bomb pattern width. However, if the target width is wider than the bomb pattern, the effective pattern width becomes the target width. The fractional target coverage is the expected fraction of the target covered by the bomb pattern given the circular error probable.

The factors which contribute to conditional damage probability include weapon reliability, number of weapons released per release pulse, number of release pulses, effective target dimensions, and effective stick pattern (Fig. 6). The weapon reliability is the probability that the weapon will mechanically operate correctly. The number of release pulses and number of weapons released per pulse are set by the pilot. The effective target dimensions include the actual length and width as well as a strip around the

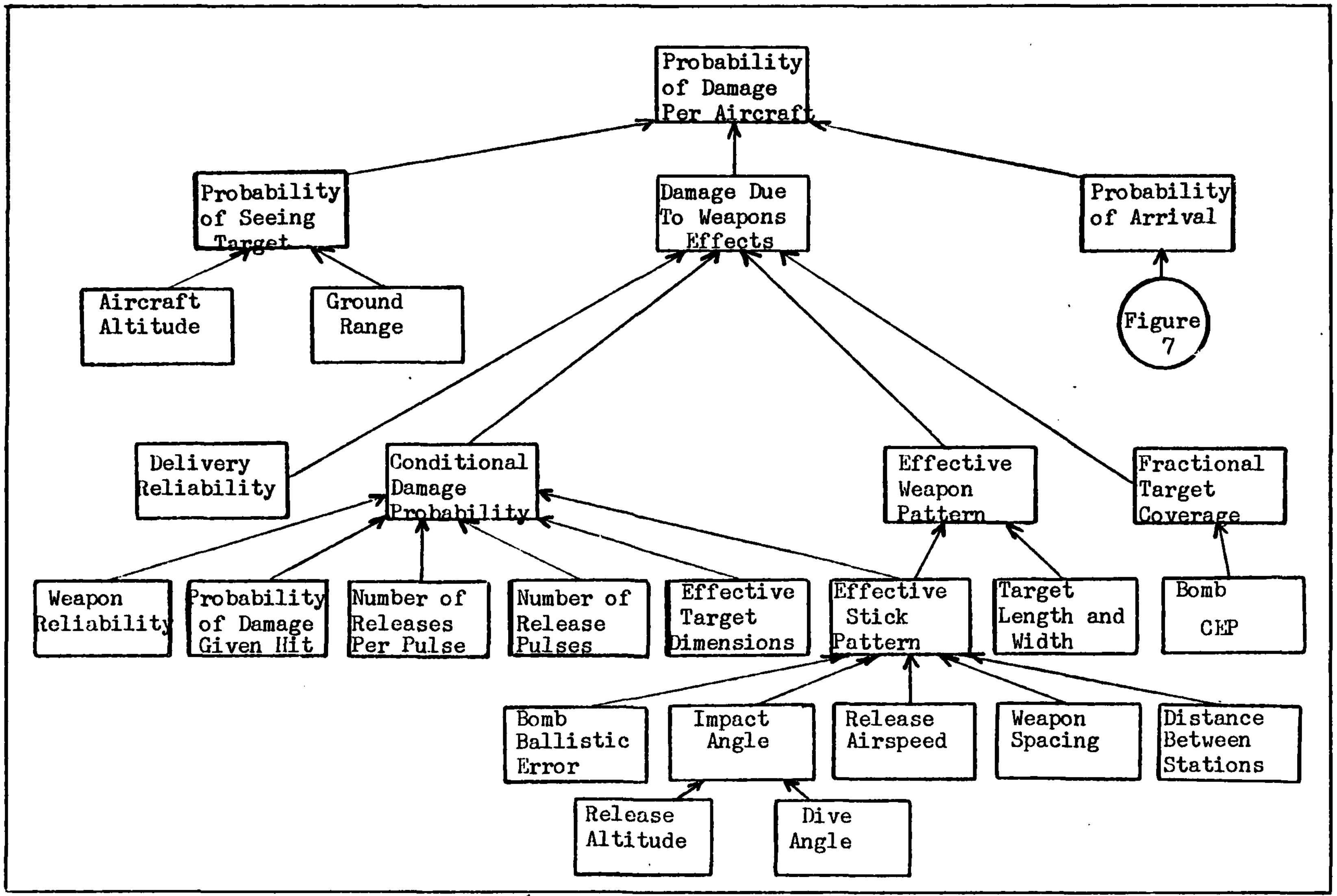


Fig. 6. Target Destruction Factors

perimeter of the target area. This strip's width is equal to the effective radius of the weapons. This accounts for the fact that weapons can impact outside the actual target dimensions and still damage the target. Finally, the effective stick pattern is the actual width of the pattern of bomb impact locations plus a strip around that pattern that again accounts for the damage radius of each impacting weapon.

The effective stick pattern is primarily a function of the conditions at which the weapons are released. The distance between the weapons stations on the aircraft, the intervolometer setting (the time between release pulses), the release airspeed, impact angle, and bomb ballistic error all determine this effective stick pattern. The impact angle for any specific weapon is a function of the altitude, airspeed, and release dive angle. The bomb ballistic error is a random variation in the bombs flight path and resulting impact point due to inconsistencies in each bomb's physical tolerances and stability characteristics. This error is commonly measured in milliradians which can be converted to feet, by the equation:

$$\text{IMPACT ERROR} = .001 \times \text{NUMBER OF MILLIRADIANS} \times \text{RANGE} \quad (5)$$

For example, a 12 milliradian ballistic error over a range of 2000 feet would result in an impact error of 24 feet.

As noted earlier, the other important aspects in determining the resultant target damage are the probability that the pilot survives with his weapons to the release point, and secondly, that he visually acquires the target prior to that release point.

The probability of visually acquiring the target is a function of the aircraft's range from the target and altitude at the time of the weapons release. In addition, it is a function of the type of terrain at the target. However, as noted earlier, the terrain around an airfield should be relatively level so the terrain effect should be minimal. Because of the increased altitude at the time of release, pop-to-low angle deliveries typically result in a higher probability of the pilot visually acquiring the target.

The other consideration, the probability of the pilot surviving, is also one of two major considerations in calculating the overall probability of survival for the aircraft.

Probability of Aircraft Survival

The factors in the aircraft's probability of survival are depicted in Figure 7. The overall probability of aircraft survival can be broken down into the probability of aircraft survival from target area entry through weapons release and the probability of aircraft survival after weapons release through exit from the immediate target area. This allows the use of the former as the probability of aircraft arrival to the weapons release point for use in

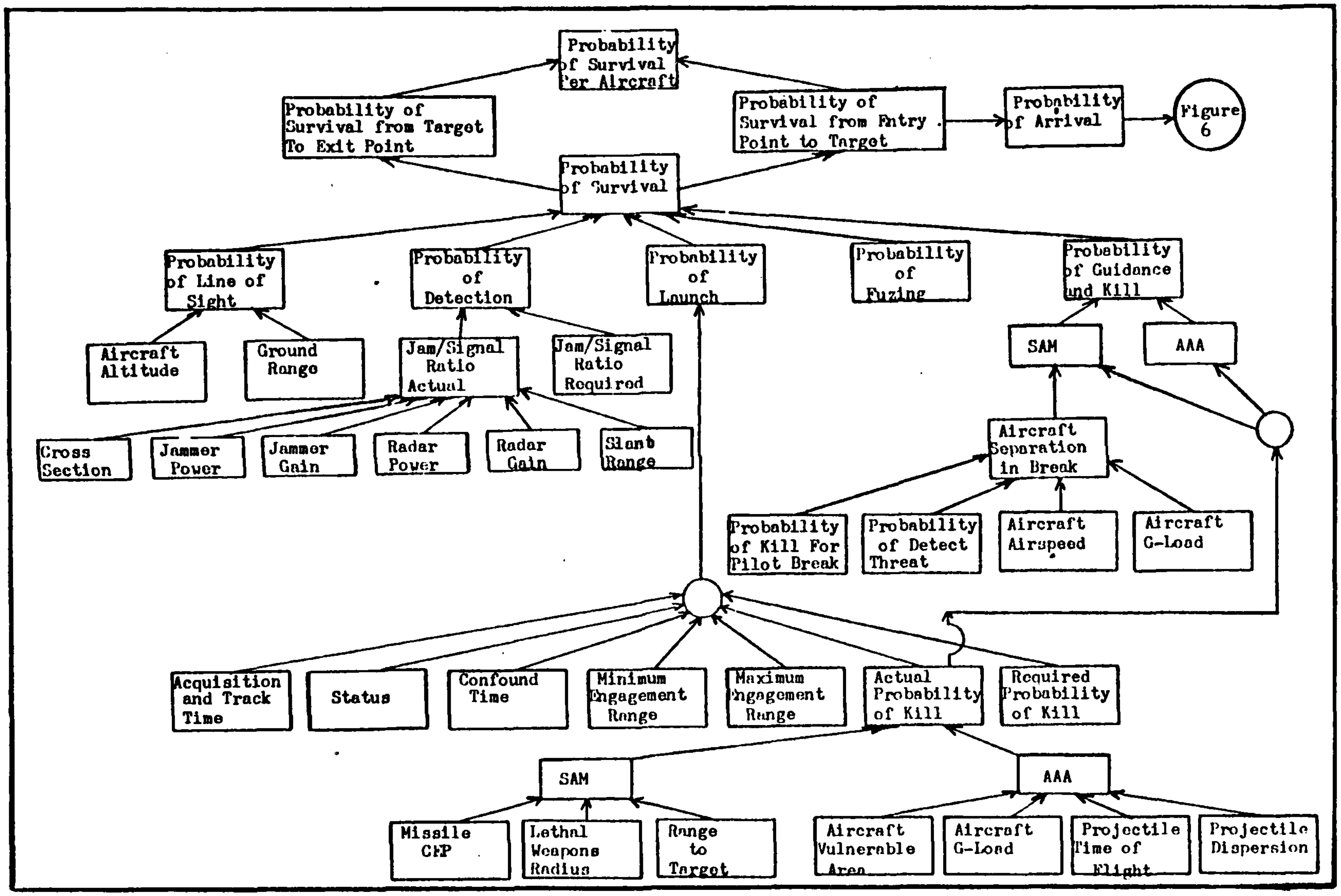


Fig. 7. Aircraft Survivability Factors

determining the probability of target damage described tactics.

Whether it is prior to the target or after the target, the probability of survival is a function of the six major factors of probability of line of sight, probability of detection, probability of launch, probability of fuzing, probability of guidance, and probability of kill given correct fuzing and guidance (Ref 4). As noted in the work by Leek and Schmitt, the probability of a clear line of sight between the threat site and the aircraft is a function of type terrain, aircraft altitude, and ground range from the site to the aircraft (Ref 14).

<u>Probability of Detection</u>. The probability of detection, given a clear line of sight, depends upon whether or not the aircraft has an effective jamming capability against the threat site radar. If the aircraft is not jamming, the ability of the threat site to detect the aircraft is a function of the returned target signal versus the clutter signal. Golden (Ref 8:44) defines this signal to clutter as:

$$S/C = \frac{Pr\ Gr^2\ \sigma\ \lambda^2}{(4\pi)^3\ R^4\ C} \qquad (6)$$

where S = returned signal from target aircraft
C = signal from the background clutter
Pr = power of threat radar transmitter
σ = radar cross section of target aircraft
Gr = gain of threat radar
λ = wave length of radar energy
R = distance from the aircraft to the threat site

By substituting the signal to clutter ratio required by the particular threat radar system, a detection range can be determined.

In the case of a target aircraft using repeater jamming, the maximum detection range can also be determined (Ref 8:125). However, the detection capability becomes a function of the jamming signal to returned aircraft signal and is defined by the following relationship:

$$J/S = \frac{P_j\ G_j\ 4\pi\ R^2}{P_r\ G_r\ \sigma} \qquad (7)$$

where P_j = output power of the aircraft jammer
G_j = gain of the aircraft jammer
R = range between the aircraft and the site radar
P_r = output power of the threat radar
G_r = gain of the threat radar
σ = radar cross section of the aircraft

Again, by substituting in the jammer's performance and the threat radar's capabilities, including the minimum jamming to signal ratio required for detection, the maximum detection range can be determined.

In addition to being within the maximum detection range, the aircraft must be at an elevation angle, α, above the horizon such that a SAM system can have a clear line of sight to the aircraft (Ref 14:13). Known as the multipath angle, this angle is defined as follows:

$$\alpha = \text{ARCSIN}(\text{ALT}/\text{SR}) \qquad (8)$$

where ALT = aircraft altitude
SR = slant range from radar site to the aircraft

Anderson and Nenner approximate this angle as .25. Therefore, given a multipath angle in excess of .25 and range inside the maximum detection range, the threat radar can detect the target aircraft.

Probability of Engagement. The assessment of the probability of engagement of SAM and AAA systems against a low altitude target aircraft is a function of site status, confound time, acquisition and track time, minimum engagement range, maximum engagement range, and engagement doctrine. The site status indicates the site's capability to be tasked against the target aircraft. For example, if the site is currently engaged against another target, that site may not be capable of engaging a second target. In addition the site may be unable to engage as expended weapons are being replaced. Anderson and Nenner explained that the confounding delay is the time it takes a site to transition to an acquisition mode once it has terminated a tracking mode on a previous aircraft. The acquisition and tracking time is the time required by the site to sufficiently acquire and track the target to determine if the targets flight path warrants an engagement (Ref 3:40). Advance target information from early warning radars can significantly reduce the time required for acquisition and track (Ref 16:29).

The minimum and maximum engagement ranges vary by the type of threat system. The parameters are a combination of weapon and radar capabilities (Ref 16:54). In the case of the AAA system, a minimum range is typically an insignificant distance.

Given a target which has been detected by an available site the launch doctrine is the decision whether or not to engage the aircraft. Anderson and Nenner comment that the threat site's decision to engage an aircraft in the FEBA is a function of anticipated probability of kill (Ref 3:52). In the case of the point defense of an airfield, it would seem more likely that a threat system would fire or launch its weapons given any probability of successful kill. That is, given the limited number of target aircraft, the high value of the airfield facilities, and the relative short period of time available to engage a target aircraft from detection until the aircraft releases its weapons, the doctrine would probably be to launch or fire the site's weapon given any probability of success.

The anticipated probability of killing the target aircraft required for the decision, regardless of the specific doctrine, can best be addressed by treating SAMs and AAA separately.

For SAM systems, once launched, the missile's probability of kill depends upon the missile's fuzing, and guidance properties as well as the lethal effects of blast and fragmentation. The guidance errors can be characterized by the missile's circular error probable (CEP) about the intended impact point. This impact point is the point in space where the aircraft and missile are forecast to collide. The CEP lies in the plane of the encounter. This plane contains the impact point and is perpendicular to the missile's flight path. The use of proximity fuzes and the lethal radius of the missile compensates for near misses. In the case of an aircraft that is jamming the threat radar with a repeater, the CEP can be calculated as follows (Ref 3:43):

$$CEP = [A(J/S)R^2 + B(J/S) + C]^{1/2} \tag{9}$$

where A, B, and C = constants which depend on the type of SAM
R = slant range from site to impact point in meters
J/S = jamming to signal ratio
CEP = circular error probable in meters

If, on the other hand, the aircraft has no jamming capability against the SAM site, the solution for CEP becomes:

$$CEP = [D(R^6)/\sigma^2 + E(R^4)/\sigma^2 + F]^{1/2} \tag{10}$$

where D, E, and F = constants for type of SAM
R = range site to impact point in meters
σ = target aircraft radar cross section in square meters
CEP = circular error probable in meters

Values of these constants for the missiles used in this research are contained in Table I (Ref 3).

The probability of accurate fuzing is the probability that the missile may detonate prior to or after the intended engagement plane. Fuzing error, therefore, when combined with guidance error, results in a three-dimensional error or volume of possible detonation points about the intended impact point.

The concept of guidance and fuzing is incorrectly accounted for in the work by Anderson and Nenner. In their work, they use CEP to represent "a sphere around the target aircraft within which 50 percent of the missiles fired under a given set of conditions will detonate." (Ref 3:42). Circular error probable is a two-dimensional, not a three-dimensional concept and applies to guidance. Any fuzing error does result in a third dimension being added to the detonation error, but it cannot be defined in terms of CEP alone.

The effects of blast and fragmentation are sometimes combined into a single measure called lethal radius. This single measure is a spherical approximation of the missile's combined fragmentation and blast capabilities given all possible encounter conditions. Specifically, this radius is the distance from the missile's detonation point where as many aircraft survive as are killed beyond it (Ref 3:42). As

TABLE I

THREAT PARAMETERS

	SAM 1	SAM 2	AAA
Minimum Altitude (Feet)	100.	100.	0.
Minimum Range	6685.	13389.	0.
Maximum Range (Feet)	33456.	72980.	9807.
Average Velocity (Feet/Second)	1722.	1965.	---
Maximum Time of Flight (Seconds)	19.4	37.1	---
Lethal Radius (Feet)	72.	86.	---
Track and Acquisition Time (Seconds)	10.	17.	6.
Confounding Delay (Seconds)	30.	30.	30.
Power Radiated (Watts-db)	50.0	53.0	50.9
Radar Gain	43.	41.	40.
Effective Radiation Power (ERP) (Watts-db)	29.6	29.6	32.3
CEP Constants			
A	325.E-9	710.E-9	---
B	1890.	2200.	---
C	25.	58.	---
D	809.E-29	403.E-29	---
E	484.E-18	102.E-18	---
F	25.	58.	---

a result, it is a cookie-cutter type of concept in that a kill is assessed if an aircraft lies within the lethal radius, and no kill is assessed if the aircraft lies outside the radius.

This concept of lethal radius will be used in this research as it has been in the previous efforts of Anderson and Nenner as well as Kizer and Neal. However, it must be emphasized that this radius is a simplifying approximation. A more detailed analysis would require the type of investigation described by Breuer (Ref 4). In such an analysis, blast and fragmentation effects would be addressed individually. In addition, while a cookie-cutter type approach is an effective measurement of blast effects, a truly accurate assessment of fragmentation effects would require the determination of a vulnerable volume around the aircraft and the calculation of the equation:

$$PKFRAG = (PKHIT)(PKGUID)(PKFUZ) \tag{11}$$

where PKFRAG = probability of kill due to fragmentation effects

PKHIT = probability of a kill due to fragmentation given a detonation within the vulnerable volume

PKGUID = probability of guidance through that volume

PKFUZ = probability of fuzing within that volume

Once it is recognized that lethal radius is a simplifying assumption, and if the fuzing error is assumed to be small, then the equation in Anderson and Nenner can be used to determine the probability of kill (PK) as a function of lethal radius and guidance error as follows (Ref 3:42):

$$Pk = 1 - (.5)^{(LR/CEP)^2} \qquad (12)$$

where LR = lethal radius
CEP = circular error probable

In the case of AAA, the probability of kill can be determined by the following calculations (Ref Kizer:58-60):

$$Pk = 1.0 - (1.0 - PKSS)^{N} \qquad (13)$$

where N = number of rounds fired in a burst
PKSS = single shot probability of kill

The PKSS can be determined by:

$$PKSS = \left(\frac{Av}{2\pi\sigma^2+Av}\right)\exp\left\{-.5\left[\frac{((9.8)(g)(TOF^2))^2}{2\pi\sigma^2+Av}\right]\right\} \qquad (14)$$

where g = the aircraft G loading
Av = average fighter aircraft vulnerable area (assumed=5.17m²)
σ = radial dispersion of bullets about the aimpoint
TOF = time of flight of the projectile

The σ can be determined by:

$$\sigma = \sigma m\ R \qquad (15)$$

where σm = mil dispersion (assumed to be 20 mils)
R = slant range from site

The TOF can be computed from the equation:

$$TOF = \frac{2m}{\rho C_d A}\left[\frac{1}{V_f} - \frac{1}{V_i}\right] \qquad (16)$$

where TOF = time of flight (seconds)
ρ = air density (assumed 1.225 kg/m^3)
Cd = drag coefficient (assumed .38 for AAA)
A = cross sectional area (assumed .0004155m^2)
m = mass of projectile (assumed .195kg)
Vi = initial velocity of projectile (assumed 930 m/sec)
Vf = final velocity of projectile (m/sec)

The Vf can be determined by:

$$V_f = V_i\, e^{-(\rho C_d A R/(2m))} \qquad (17)$$

where Vf = final velocity of projectile (m/sec)
R = slant range from aircraft to site (kilometers)
other values defined previously

Kizer and Neal noted that by using the assumed value for the parameters in the AAA equations, the TOF is determined by:

$$TOF = 2014.46\left[\frac{1}{V_f} - \frac{1}{V_i}\right] \qquad (18)$$

Again, the threat site will determine the forecasted probabiliy of kill prior to launch. This probability of kill, when combined with the other factors of site status, confound time, acquisition and track time, and minimum/maximum engagement ranges will be assessed and a launch decision will be made according to the launch doctrine.

Probability of Guidance/Kill. The probability of guidance and kill is the final set of factors used to assess the probability of an aircraft's survival.

If the aircraft does not maneuver after threat site launch, the probability of kill at the completion of the missile or AAA projectile intercept will be the same as the probability of kill obtained in the launch calculations. However, if the aircraft maneuvers during the missile or bullet time of flight, the end of intercept CEP and resulting PK can vary.

One reason for this change in Pk in the SAM engagement is that the impact point will change from that planned at launch. The range from the new impact point to the threat site will be used to determine the new CEP and new Pk. In addition, Leek and Schmitt pointed out that the aircraft's profile to the site will change during an aircraft maneuver. The result will be a change in aircraft radar cross section and resulting change in CEP and Pk (Ref 14:19).

Another factor in the SAM engagement which has not been addressed in previous theses is the affect of an aircraft actively maneuvering to defeat the missile. Previous theses assumed that given an aircraft maneuver, a new impact point could be calculated and the missile redirected to that impact point. In these cases, given the previous comments on lethal radius, the use of the equation:

$$Pk = 1 - (.5)^{(LR/CEP)^2} \qquad (12)$$

is satisfactory as the assumption is made that the missile and aircraft will still be aimed at the common, updated impact point. However, if the pilot performs a "SAM break" against the missile, the missile's maneuvering ability is not normally sufficient to alter its course sufficiently to arrive at the new impact point dictated by the aircraft's maneuver. As a result, at missile detonation time, the aircraft will not be at the predicted impact point of the missile. In addition, the aircraft will probably not be in the plane used to calculate the effect of missile CEP. It is, therefore, inappropriate to use the equation:

$$Pk = 1 - (.5)^{(LR/CEP)^2} \qquad (12)$$

which assumes only a two-dimensional relationship. Instead, a three-dimensional relationship must be used. Therefore, it appears a more appropriate method to calculate probability of kill in this case could be via a cell model as described by Bridgeman (Ref 5).

In the cell model, the CEP plane, the plane that contains the missile's impact point and is perpendicular to the missile's flight path, is divided into cells. Figure 8

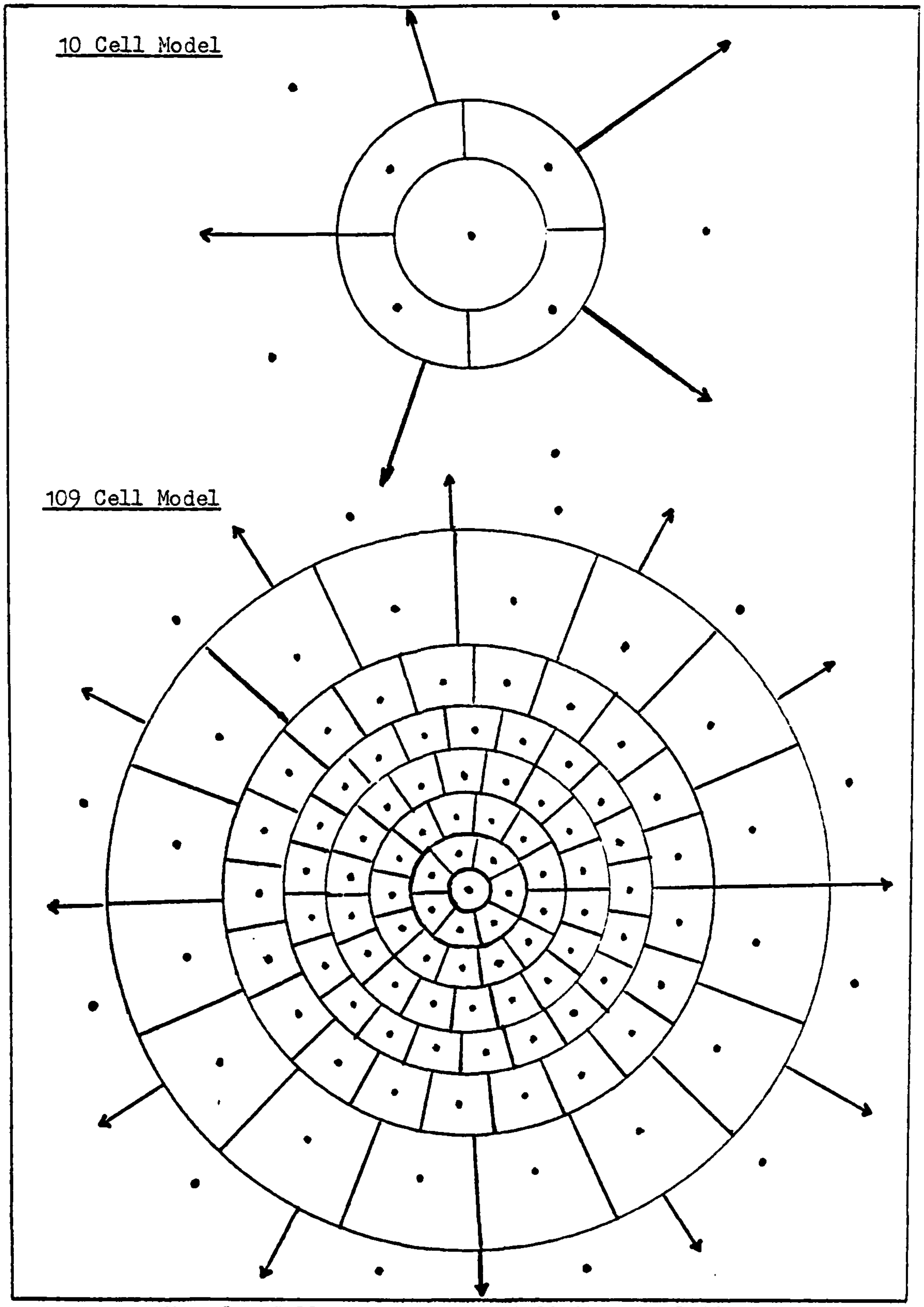


Fig. 8. Cell Structure for Cell Pk Calculations

depicts a 10 cell and 109 cell model. These cells represent all possible impact points from directly on the desired impact point out to an infinite distance away from the desired impact point in the CEP plane.

The center of each of these cells is defined in terms of a radial distance and angle. The reference for the angle is the line between the center of the network of cells and the projection of the target's position into the plane. The radial distance to the cell center is a product of the missile CEP and a dimensionless adjustment factor. In the case of a ten cell model the values of these adjustment factors are:

CELL 1	= .0000
CELL 2 to 5	= .7109
CELL 6 to 10	= 1.5090

In the case of a 109 cell model, the values of these factors are:

CELL 1	= 0
CELLS 2 to 8	= .2506
CELLS 9 to 21	= .4722
CELLS 22 to 39	= .7162
CELLS 40 to 60	= .9950
CELLS 61 to 81	= 1.3300
CELLS 82 to 97	= 1.8170
CELLS 98 to 109	= 2.5370

The distance from each cell center to the aircraft position at the time of impact is determined and compared to the missile's lethal radius. If the distance is less than the lethal radius, the probabiliy of kill given a missile impact in that cell equals one. The probability of killing the aircraft for an impact in that cell equals zero if the cell center to aircraft position distance exceeds the lethal radius. The probability of the missile actually landing in the cell is then multiplied times the probability of kill for that cell in order to determine the cell's contribution to overall probability of kill. (The probability of the missile landing in a cell is approximately 10% for the 10 cell model and 1% for the 109 cell model.) The probability of this SAM killing the aircraft then becomes the sum of the individual cell contributions to overall probability of kill.

In the case of AAA, if the aircraft maneuvers during the time of flight of the projectiles, the cell model is more appropriate than the equations for AAA probability of kill noted earlier. However, if the time of flight of the AAA is short, or if the aircraft does not actively maneuver against the AAA during its time of flight, the equations are appropriate.

Overall Mission Damage and Survivability

The overall target damage inflicted and the aircraft survivability of the mission of two aircraft can be determined by combining the individual probabilities of each aircraft. The target damage as a result of the mission of two aircraft can be obtained from the equation:

$$PDMSN = 1 - (1-PTGTDMG1)(1-PTGTDMG2) \qquad (19)$$

where PDMSN = probability of target damage by the mission of two aircraft.
PTGTDMG = probability of target damage due to one aircraft number (1) or two (2).

The overall probability of aircraft survival for the mission of two aircraft is determined by the relationship:

$$PSMSN = (PS1)(PS2) \qquad (20)$$

where PSMSN = probability that the flight of two aircraft survives the mission.
PS = probability of survival for aircraft number one (1) or two (2).

Summary

In order to predict the level of target damage and the corresponding aircraft survival for an offensive counterair mission, four basic processes should be performed. First, the attack planning should consider the type of routing, ingress tactic, delivery and weapons suitable in view of the threats, terrain, weather and target characteristics. Second, the probability of target damage from a single aircraft should be addressed. Third, the aircraft probability of individual aircraft survival should be determined. This probability is a function of the threat site's combined probabilities of line of sight, detection, launch, fuzing, guidance, and kill. Finally, the individual aircraft probabilities of target damage and aircraft survival should be combined to form an overall mission probability of target kill and aircraft survival.

III. Research Model

Thierauf and Grosse point out that a model is a representation of an actual event or situation. As such, the model shows the interactions of multiple variables in a cause-and-effect relationship. As well as investigating the relationships among variables, these authors note that the model can be used to determine which variables are most important. In this process it is critical that the model be a good representation of the real world events which it portrays (Ref 22:14-15).

The model for this research must, therefore, identify the interactions that result in target damage and aircraft survival. The calculation of effects of weapons impacting on the target can be modeled analytically via the logic contained in JMEM. The authors of JMEM note that the JMEM logic accounts for variations from the desired release parameters. In other words, the logic results in average outputs for the specific input variables.

As noted earlier, the JMEM logic does not account for the probability of the aircraft arriving at the release point nor the probability of the pilot seeing the target at the release point. The probability of the aircraft surviving the defense systems, releasing weapons, and safely exiting the area is a complex dynamic problem. Many of the critical variables, such as aircraft position and slant range to threats change continuously over time. This indicates that a purely analytical solution of the survivability question is inappropriate.

Since a pure analytical approach is not suitable, simulation was considered as an alternative. Hillier and Lieberman point out that simulation is costlier than analytical methods and fails to yield a specific answer. However, they also note that simulation "is an invaluable tool for use on those problems where analytical tools are inadequate" (Ref 10:672-673). Shannon notes that in addition to being an invaluable tool when analytical techniques are inappropriate, simulation can provide a time history of the modeled process (Ref 19:11). As a result of these considerations, simulation was chosen as the method to route the aircraft through the target area.

Another consideration for the model was the type of computer language to use. Pritsker and Pegden describe different types of simulation systems which may dictate that certain languages be used. These authors describe a discrete simulation system as one in which variables change only at specific points in time. A continuous simulation system, on the other hand, contains variables which change continuously over time. Finally, a combined discrete-continuous model is one in which the variables may change both discretely and continuously. They note that an excellent, powerful language to use in such cases is SLAM (Ref 17:62,74).

The scope of this research is characteristic of the discrete-continuous model. The relationship of the aircraft with other entities in the system is continuously changing over time. An example of this would be the changing slant ranges between the aircraft, the threats, and the aircraft's intended navigation points. Some of the relationships are discrete and can be scheduled, such as the desired time that each aircraft enters the target area or the time a missile is scheduled to be fired.

As a result of these considerations, this research incorporates a simulation model. The JMEM logic for probability of target damage due to weapons effects is modeled via Fortran inserts into a SLAM simulation model.

This research does, however, minimize its dependence on the SLAM simulation structure. The continuous modeling capabilities of SLAM are used only to "fly" the aircraft through the target area and to react to threats. SAMs are not flown continuously to their targets. Rather, the movement of each SAM is discretely scheduled. To further minimize the understanding of SLAM required to operate this model, the discrete events and file structures typical of SLAM discrete modeling are to be used. Discrete relationships are modeled using scheduling concepts available in SLAM continuous modeling. Standard Fortran arrays are used to store data, and Fortran "if" statements are used to schedule a few events which occur only once during the entire simulation run. The stochastic relationship that may arise, such as the probability that a pilot is aware of a missile attack, can be modeled via the distribution function which are available in SLAM continuous modeling.

Assumptions

Given the scope and scenario described previously this model development incorporates the following assumptions:

1. Aircraft airspeed is a constant 525 knots the entire time the aircraft is within the 10 nautical mile radius target area.

2. The pilot will update his position prior to entering the target area. The accuracies of his heading and navigation systems will allow him to reach each of his planned navigation points.

3. A 30 degree angle off pop-up attack will be planned for all angular deliveries.

4. A pilot who still has his bombs at the target will release all weapons on a single pass.

5. The POL tanks are uniformly distributed within the defined target dimensions.

6. The tracking time for each delivery is five seconds prior to the release of the first weapon.

7. The CEP for level deliveries is 250 feet and for angular deliveries is 125 feet.

8. When the target of a SAM launch, the pilot will be able to predict SAM probability of kill and time to impact which is comparable to the information possessed by the threat site and missile.

9. The aircraft will not maneuver to counter the short

AAA burst.

10. At time to impact of two seconds or less, a pilot who is aware of a SAM whose probability of kill exceeds the probability of kill required for the pilot to evade the missile will evade the missile regardless of his phase of flight.

11. A pilot who decides to evade a missile will jettison his weapons, attempt to defeat the missile, and depart the target area.

12. A pilot maneuver against a SAM will be perfect. That is, the missile will be unable to modify the predicted impact point in the missile aircraft plane once the pilot has begun his "SAM break".

13. Any pilot who is not aware of a SAM threat, decides not to react to a SAM threat, or is engaged by an AAA threat will continue his planned routing through the target area.

14. The only threat systems modeled will be AAA and SAM.

15. An aircraft above 100 feet within the target area (ten nautical mile radius) satisfies the multipath angle of all threats and can, therefore, be acquired by any threat radar that is not currently engaged.

16. Each threat site is capable of engaging only one aircraft at a time.

17. Computations involving aircraft radar cross section will be based on an average cross section of 2.5 square meters.

18. The minimum acquisition and missile engagement altitude for SAM systems is 100 feet above ground. All other missile parameters are as defined in Neal and Kizer's work and listed in Table I (Ref 16:55-56).

19. Each SAM site has two missiles which can be launched during the airfield attack. These missiles may be launched against a single aircraft or split between the two aircraft.

20. Each SAM uses collision (proportional navigation) steering to intercept an aircraft.

21. A SAM system can continue to track an aircraft inside of minimum missile range.

22. A SAM system will fire a first missile at its targeted aircraft one second after track and acquisition time has elapsed, provided the impact point is within the minimum and maximum ranges of that type SAM.

23. The missile's maximum time of flight (TOF) is the time to fly a distance equal to the maximum range of the threat site.

24. After launch, each missile will fly a constant speed as defined in Table I.

25. A SAM system will fire a second missile (if available) at its targeted aircraft a minimum of ten seconds after the launch of the first missile and as soon as the predicted impact point of the second missile is within the site's minimum and maximum ranges.

26. The fuzing error for each missile's warhead is zero.

27. A probability of kill of zero will be assessed to any SAM whose targeted aircraft reaches the target area's 10 nautical mile ring.

28. Due to "cool down" limitations, an AAA site can only fire a single, one second burst of projectiles against each aircraft.

29. An AAA system will fire the single burst of 100 rounds at its targeted aircraft as soon as the aircraft enters the AAA maximum range and the six second tracking delay has elapsed.

Model Overview

The model in this research consists of three major parts. The first is subroutine INTLC which is used to set the initial conditions prior to the actual start of the simulation. The second major section is subroutine STATE which contains the dynamic difference equations used to define state variables, SS(I). the third division is subroutine EVENT and its associated Fortran subroutines. Subroutine EVENT is used to discretely change the values of global variables, XX(I), state variables, or user defined variables in response to the occurrence of scheduled time or state events. Each of these three major portions of the model will be addressed in greater detail after an explanation of the coordinate system which is used in this model.

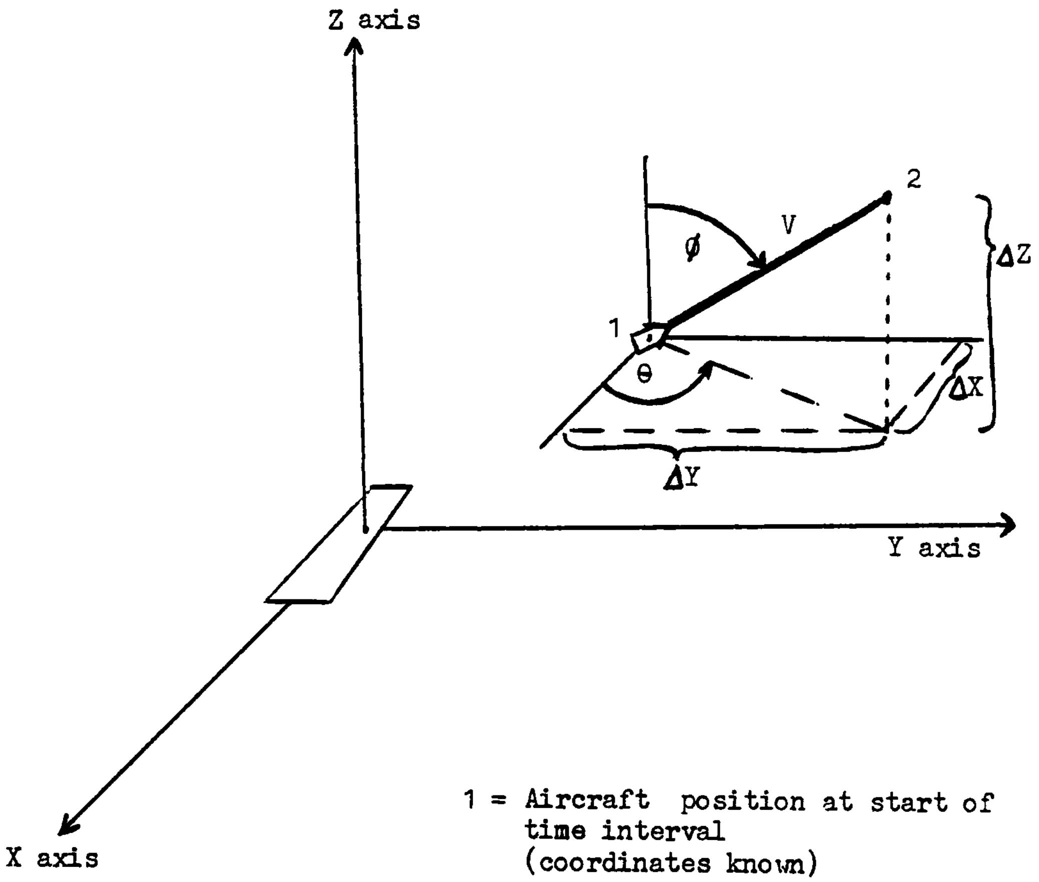
Coordinate System

The model is based on a three-dimensional cartesian reference system centered on the center of the airfield's runway. Since direction of the aircraft movement is a function of its pitch and heading, it is necessary to transform headings and pitch into spherical coordinate angles Φ and Θ. As depicted in Figure 9, the changes in aircraft x, y, and z coordinates during any time increment, given the heading, pitch angle, and velocity of the aircraft from some known point are:

$$\Delta x = \rho \text{ Sin } \Phi \text{ Cos } \Theta \tag{21}$$
$$\Delta y = \rho \text{ Sin } \Phi \text{ Sin } \Theta \tag{22}$$
$$\Delta z = \rho \text{ Cos } \Phi \tag{23}$$

where ρ = distance aircraft traveled during time step
Φ = angle due to pitch
Θ = angle due to heading
$\Delta x, \Delta y$, and Δz = changes in x, y, and z coordinates during the time intervals.

Since aircraft heading is oriented to north and increases clockwise from 0 degrees to a maximum of 360 degrees, it is necessary to convert, aircraft heading into the Θ of the spherical coordinate system. Likewise, since aircraft pitch equals 0 degrees when parallel to the ground and increases to +90 degrees up or decreases to -90 degrees down, it is necessary to convert the pitch angle to Φ degrees. The



1 = Aircraft position at start of time interval (coordinates known)

2 = Aircraft position at end of time interval (coordinates unknown)

Fig. 9. Coordinate System

following relationships are used to perform these conversions:

$$\phi = 90 - PCH \quad (24)$$
$$\theta = 360 - HDG + 90 \quad (25)$$

where PCH = aircraft pitch angle above or below horizon
ϕ = aircraft pitch angle in spherical coordinate system
HDG = aircraft heading in ground plane
θ = aircraft heading in spherical coordinate system

Subroutine State

In continuous simulation modeling, subroutine state is used to define changes in state variables as a function of time. The subroutine is executed at least once each time increment to update the values of the state variables via difference equations. The values of these state variables are then used throughout other parts of the program to redefine relationships between other variables. In this model for instance, state variables are used to model the x, y, and z position of each aircraft in the three-dimensional coordinate system centered on the enemy runway. In addition, heading and pitch angle of each aircraft are defined as state variables. State variables are also used to define current ground range from each aircraft to each of the threat sites and the runway center, and the next navigation point. In addition, the current system time and the time the pilot completes his post release maneuver are represented with state variables.

In addition to updating state variables, this model uses subroutine STATE to schedule several events via Fortran "if-then" structures. Each of these events is scheduled only once. The logic for each event depends on the value of one of the previously mentioned state variables.

An important aspect of SLAM continuous simulation used in this model is the ability to change accelerations over time. These accelerations are the rates and directions which state variables such as heading and pitch change. Therefore, in addition to establishing the magnitude of an acceleration, this model uses pitch and heading direction flags to indicate whether the acceleration is up or down for pitch or if acceleration is right or left for heading.

Subroutine INTLC

The purpose of subroutine INTLC in SLAM continuous modeling is to establish initial values for variables at the beginning of the simulation run. In this model, the bulk of this task is accomplished by a Fortran call to subroutine BIGPIC. Subroutine BIGPIC is used to initialize the arrays to zero and assign initial values as a function of the entering arguments for the simulation run. In addition to the call to subroutine BIGPPIC, subroutine INTLC initializes the values of many of the global xx variables and state ss variables which are functions of values returned from the call to subroutine BIGPIC.

Subroutine BIGPIC

The purpose of subroutine BIGPIC is to initialize values which are stored in the major arrays used in the model (ACFT,TH,TGT,MSL). The array ACFT contains specific information about each of the two aircraft. Items such as tail number, planned dive angle, and coordinates for each of the aircrafts navigation points is stored in the array. The TH array contains information on each of the enemy's SAM and AAA sites. Coordinates of the threat sites, maximum range, and number of available missiles are among the items stored for SAMS in this array, but only the site coordinates are stored for AAA sites. The array TGT is a small array containing only the x, y, z coordinates of the center point of the target, the target's dimensions, and the orientation (in magnetic degrees) of one side of the target area. The last of the arrays is MSL which contains specific information about the missile including, among others, its heading, pitch, location, and maximum time of flight remaining. When the subroutine is called, all of each array's values are initialized to zero prior to the assignment of any values to the array.

Assignment of the values in the subroutine is via explicit equations as well as Fortran calls to three subroutines: SCAN4, JMEM6, and ROUTE7. These three subroutines perform threat search, weapons effects, and flight routing functions.

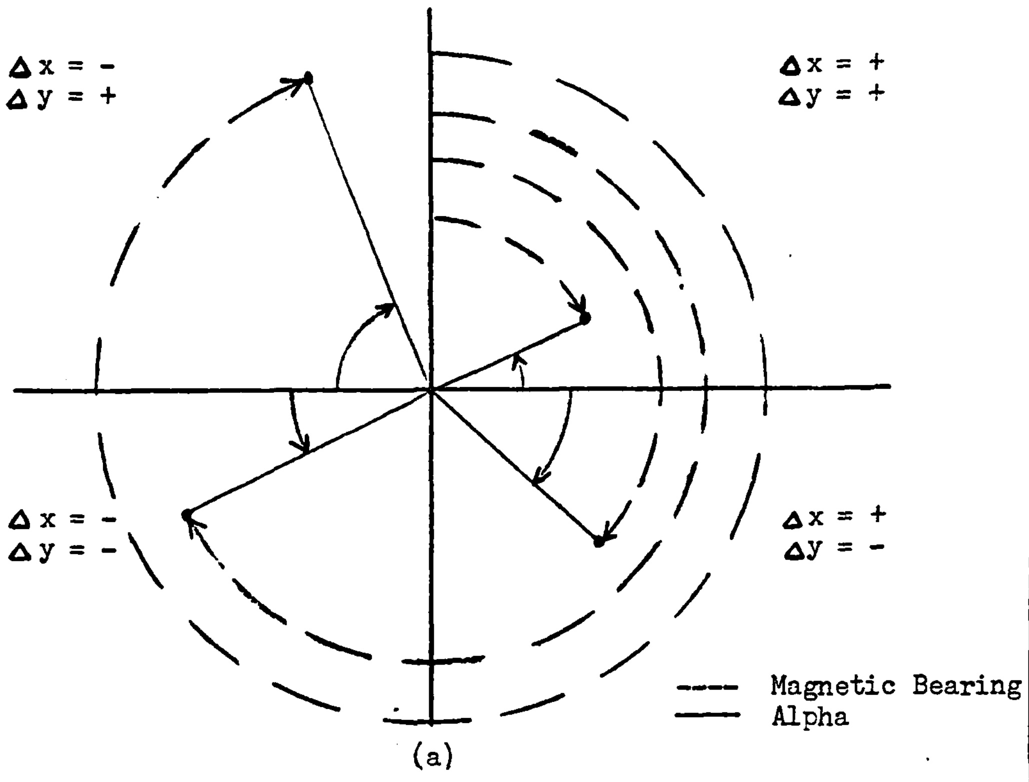
Subroutine XYCOR1

Subroutine XYCOR1 is used to determine the x and y coordinates of a point B given the x and y coordinates of another point A, the ground range from A to B, and the magnetic bearing (MB) from A to B. The MB is assigned in a clockwise direction from the vertical and the angle α between the MB and nearest x axis is determined. That is, α is always a value between 0 and 90 degrees. A depiction of this process for each of the four quadrants is contained in Figure 10(a). The magnitude of the changes in the x and y coordinates between the known coordinates of point A and the unknown coordinates of point B are determined by the following relationships:

$$\Delta x = GR * \cos \alpha \tag{26}$$
$$\Delta y = GR * \sin \alpha \tag{27}$$

where GR = ground range from point A to point B
α = angle formed by MB and nearest x axis
Δx and Δy = magnitude of change in x and y directions to move from point A to point B.

The directions of these changes are determined by the quadrant in which the MB is located.



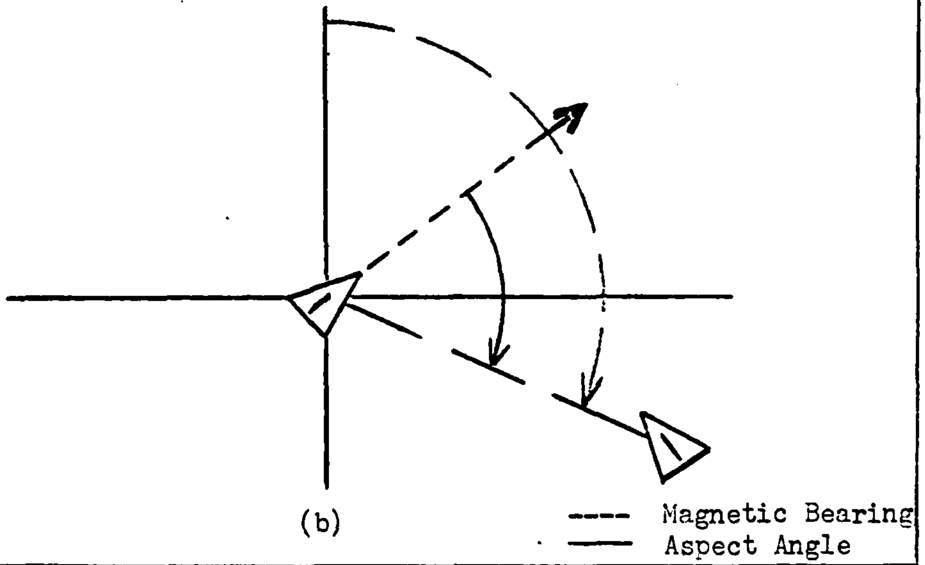


Fig. 10. Coordinate and Magnetic Bearing Geometry

Subroutine MBRAN2

The purpose of subroutine MBRAN2 is to determine the relationship of point B from point A. The input variables for the subroutine are the x, y, and z coordinates of each point, A and B. The output is the ground range (GR2), slant range (SR2), magnetic bearing (MB2) of point B from point A, and aspect angle (AA2) of point B reference an entity of point A. While MB2 is measured in degrees from directly north of the position of point A clockwise to point B, the aspect angle is measured in degrees from directly in front of an entity, such as in aircraft, at point A clockwise around the entity to point B. The relationship of these two concepts are depicted in Figure 10(b).

The method used to determine the magnetic bearing is to calculate how x and y change in moving from point A to point B as well as the angle α formed between point B and the nearest x axis. This is basically the reverse of the procedure used in subroutine XYCOR1. This angle α is then added or subtracted from a constant depending upon whether point B is north or south of point A. The constant is 90 if point B is east of point A and is 270 if point B is west of point A. The result of the angle being applied to the constant is the magnetic bearing of point B from point A. The aspect angle is obtained by subtracting the heading of the entity at point A from the magnetic bearing. If the result is negative, it is added to 360 to keep it positive. The ground range and slant range are each calculated as the square roots of the sum of the squares of the coordinate changes. Finally, given the heading at point A, the direction of shortest turn to point B is determined.

<u>Subroutine SCAN4</u>

The purpose of subroutine SCAN4 is to determine an approach axis and specific final attack headings for each aircraft which minimize exposure to enemy threats. The input variables for the subroutine are the coordinates of each threat site and the target center as well as the orientation, or centerline, of the target's length. The subroutine logic investigates 120 degree cones centered on the centerline at either end of the target area (Fig 11). The cone containing the least number of threat sites is the cone through which the aircraft will approach the target. Once the target approach is determined, the final heading for each aircraft is calculated. The subroutine assigns headings to the leader and wingman which are 20 degrees apart to avoid identical ground tracks. In addition, the subroutine assigns the lead aircraft a ground track which is between the runway and wingman's aircraft during the approach to the target. This insures the wingman can look in the direction of the lead aircraft and still have the runway's threats in his field of view. Finally, the subroutine calculates a roll-in to final direction for each aircraft. This is the direction a pilot doing a pop-to-angular delivery will turn to arrive on his final target heading. The subroutine assesses the aspect of

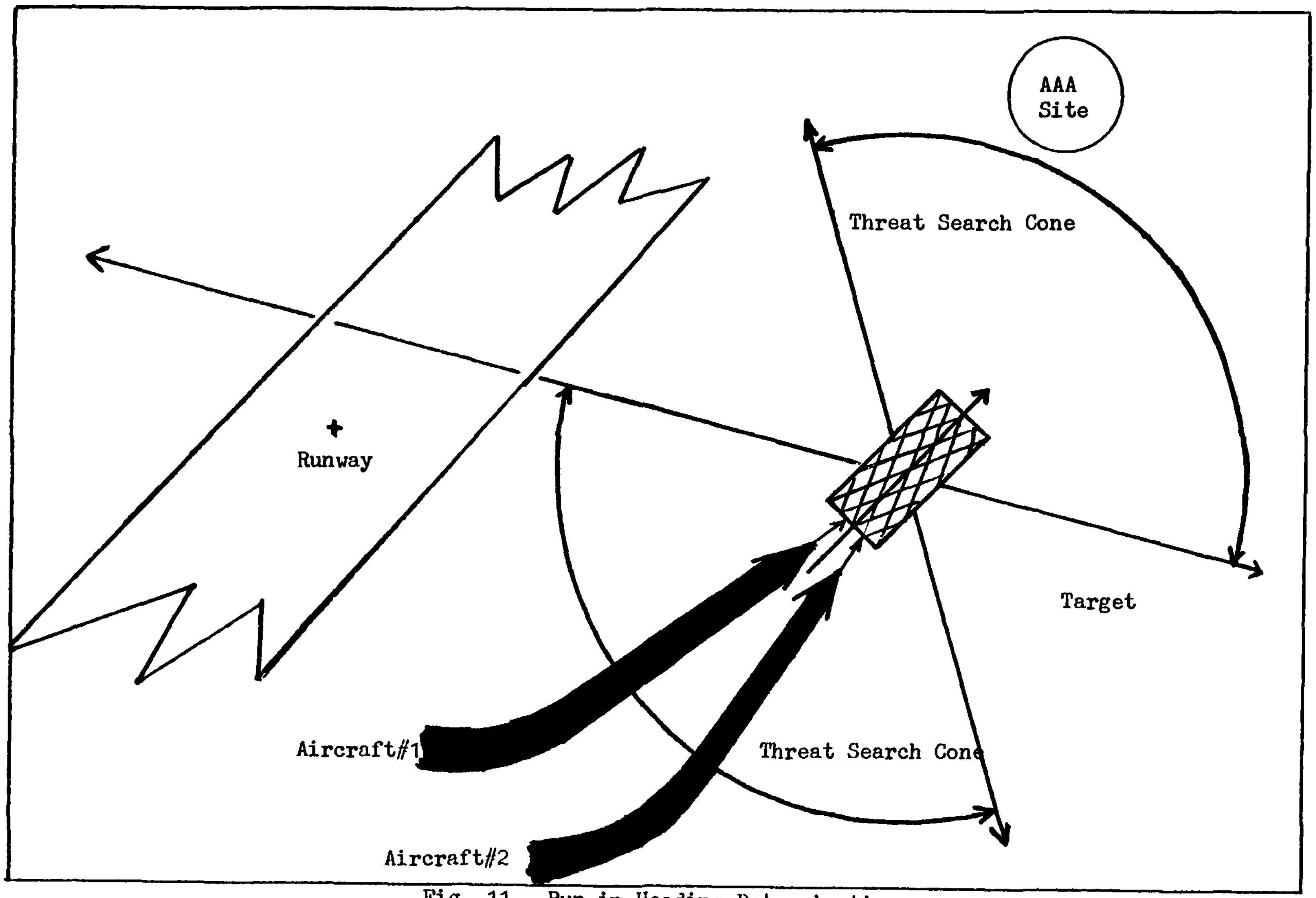


Fig. 11. Run-in Heading Determination

the runway with respect to a point five miles prior to the area target. The aircraft are then assigned turn directions which insure they are not turned away, or "belly-up" to the runway during the turn to final. This is depicted in Figure 11.

Subroutine JMEM6

The purpose of subroutine JMEM6 is to produce a probability of target damage due to weapons effects for each aircraft. The subroutine also provides data, such as bomb range which is used later by subroutine ROUTE7 for route planning. The logic in subroutine JMEM6 is based on an analytical solution which incorporates logic contained in the Joint Munitions Effectiveness Manual (JMEM) (Ref 20:Ch 4,65-66). It should be noted that the logic in the original JMEM method does not address the probability of the aircraft survival nor the probability of the pilot seeing the target from his release point.

The entering arguments for the subroutine are contained in Table II. Most of these values are entered via the initialization of the arrays ACFT and TGT in subroutine BIGPIC. The bomb ballistic errors, the weapons reliability, and the circular error probable are approximations of actual values. In addition to these entering arguments, the logic of the manual requires reference to graphs and tables to obtain the average bomb impact angle, the slant range of the first weapon from release point to ground impact point, the

TABLE II

ENTERING ARGUMENTS FOR JMEM6

Release airspeed (Knots)
Weapons type (low drag or high drag)
Release dive angle (degrees)
Target length (feet)
Target width (feet)
Bomb ballistic error (milliradians)
Weapon reliability
Total number of bombs on aircraft
Number of bomb releases per pulse
Number of release pulses
Circular error probable (feet)
Intervolometer setting (seconds)
Delivery Reliability
Distance between aircraft weapon stations (feet)
Release altitude of last weapon in string (feet)

expected fractional coverage of area, and the effectiveness type and index.

The calculation of impact angle and slant range is made in subroutine JMEM6 by the use of linear equations which approximate portions of the unclassified graphs in the manual. In order to reduce the error associated with these approximations, a limited range of release conditions is represented by the equations of this model. The equations assume a 525 knot release airspeed, a specified dive angle, and within the altitude regimes listed in the code for each delivery.

Similar to the impact angle and slant range, in the manual, the determination of expected fractional coverage of the target normally requires reference to a graph. The graph, however, is based on the equation (Ref 20:C-12):

$$E = \frac{2\sqrt{2}}{(C6)(T6)\sqrt{\pi}} \left[a6 \int_0^{a6} e^{-t^2}\, dt - b6 \int_0^{b6} e^{-t^2}\, dt + \tfrac{1}{2} \left[e^{-a6^2} - e^{-b6^2} \right] \right] \quad (28)$$

where E = expected fractional coverage in length or width
C6 = .6745
T6 = LA/DEP for range
T6 = WA/DEP for width
a6 = $C6(P6 + T6)/(2\sqrt{2})$
b6 = $C6|P6 - T6|/(2\sqrt{2})$
P6 = LEP/REP for range
P6 = WEP/DEP for width

and LA, WA, LEP, WEP, REP, DEP are as listed on the variables list in Appendix A. This model solves the equation to obtain the fractional coverage. The area under each integral is determined by a call to function AREA5 which uses a numerical integration technique to assess the areas. These values for the areas are then returned to subroutine JMEM6 and the expected fractional coverage in length and width is calculated.

Another portion of the manual's logic requires reference to a classified table to obtain the effectiveness index type and value. The entering arguments for the table are weapon type, type target, and impact angle. For this research, an approximate index type and index value are used. The type is MAEB and index value is 3000.

Finally, the manual requires reference to worksheets to determine delivery accuracy and weapons effects dimensions. The relationships extracted from these worksheets and used in this research are:

$$DEP = .573\ CEP \tag{29}$$
$$REP = CEP \tag{30}$$

where CEP = circular error probable in the ground plane, feet
DEP = deflection error probable, feet
REP = range error probable, feet

AET = MAEB
WET = MAEB
LET = WET

where MAEB = effectiveness index value
AET = effective target-element area in ground plane, feet
WET = effective target width, feet
LET = effective target length, feet

In addition to the determination of the probability of target damage due to weapons effects, subroutine JMEM6 includes Fortran logic for the probability of the pilot seeing the target. This is the likelihood that the pilot can see his target when he is at his weapons release point and altitude. The probability is a function of ground range and altitude above the target. This information is based on a graph in JMEM (Ref 21:Ch 5,6). The graph's information is not included in the analytical damage assessment method of JMEM but is included in subroutine JMEM6 to satisfy objectives of this research.

Function AREA5

As described in the description of subroutine JMEM6, the Fortran function AREA5 is a numerical integration routine for use in the solution of expected fractional coverage in length and in width of a target by the impacting weapons.

Subroutine ROUTE7

Subroutine ROUTE7 is used to plan each aircraft's route of flight, starting at the target and ending back at the entry point for each aircraft as it enters the target ten mile radius circle. The subroutine calculates the x, y, and z coordinates of each turn point as well as the pitch angle and heading which the pilot would be using as he approaches these points. An egress or post weapons release turn direction is also calculated to get the pilot headed out of the target area in the shortest time possible.

The input variables for the subroutine are included in Table III. The heading for each aircraft and roll-in direction are passed via Common Statements from subroutine SCAN4 to subroutine ROUTE7. The rest of these entering

TABLE III

INPUTS FOR SUBROUTINE ROUTE7

Lead aircraft's heading against target, degrees
Wingman aircraft's heading against target, degrees
Roll-in direction on pop maneuver
Tail number of aircraft
Aircraft planned release dive angle, degrees
Aircraft planned release velocity, knots
Aircraft ingress altitude, feet
Ground range from first release point to target, feet
Release altitude of first weapon, feet
Release altitude of last weapon, feet
Aircraft intervolometer setting, seconds
Number of release pulses

values are passed to the subroutine via the array ACFT. With these initial values, the logic in subroutine ROUTE7 allows calculation of each plane's navigational information and storage of this information in array ACFT.

Most of the coordinates for the points are determined via a call to subroutine XYCOR1. The arguments passed to subroutine XYCOR1 are the ground range and magnetic bearing of the unknown point from the navigation point whose coordinates have most recently been calculated. An exception to this occurs in the roll-in maneuver for the pop-to-angular deliveries. The specific details of how ground range and magnetic bearing for each point is calculated will now be addressed.

The location of the first weapon release point is found by providing subroutine XYCOR1 a ground range equal to the ground range from first release point to the target obtained in subroutine JMEM6. The magnetic bearing is the run-in heading minus 180 degrees. The subroutine contains logic to insure that the magnetic bearing, any aspect angles, and headings remain within a 0 to 360 degree heading system. The anticipated pitch is the desired dive angle from array ACFT and the anticipated heading is the run-in heading.

The coordinates of the last release point are calculated by reference to these coordinates of the first weapon release point. The magnetic bearing uses the run-in heading. The ground range is obtained from the following:

```
Level Delivery:
 GR = V7*1.689*ACFT(i,6)*((ACFT(i,9)/ACFT(i,5))-1) (31)


Angular Delivery:
 GR = (ACFT(i,12)-ACFT(i,4))Sin((90-ADIV7)/57.3)/  (32)
      Sin(ADIV/57.3)


 where  GR = ground range traveled  between release of
             first and last weapon
        V7 = aircraft velocity in knots
 ACFT(i,6) = intervolometer setting in seconds
 ACFT(i,9) = total number of weapons on aircraft
 ACFT(i,5) = number of bombs released per pulse
      ADIV = aircraft dive angle
```

The coordinates for the track point are found via a ground range and magnetic bearing which is referenced to the first weapon release point. The magnetic bearing equals the runin heading minus 180 degrees. The ground range is obtained by the relationship:

```
        GR = TRKTM*1.689*V7*Cos(ADIV/57.3)          (33)

 where  GR = ground range from track point to first
             weapons release point
        V7 = aircraft velocity
     TRKTM = tracking time prior to release
     ADIV7 = aircraft dive angle
```

The planned dive angle and heading at the track point are the planned dive angle and run-in heading.

The model then investigates whether the aircraft is scheduled to perform a pop-to-angular delivery or a level delivery. If the delivery is a pop-to-angular, the roll-in point is obtained via the geometric relationship portrayed in Figure 12. The turn to final is based on a 4 G turn in the horizontal plane. The radius of this turn, POPRAD is:

$$POPRAD = (V7*1.689)^2/(32.2*G) \tag{34}$$

where V7 = aircraft velocity in knots
G = aircraft G load of 4 G's

The distance on the arc, the SARC, during the turn is:

$$SARC = (DEGTT/57.3)\ POPRAD \tag{35}$$

where DEGTT = degrees to turn (30 degrees for this research)
POPRAD = radius of turn, feet

Each of the angles A can then be obtained with:

$$A = (180-DEGTT)/2.0 \tag{36}$$

The distance, HYPOT, between the roll-in and track points is then calculated as:

$$HYPOT = (POPRAD)(Sin(DEGTT/57.3))/(Sin(A/57.3)) \tag{37}$$

This HYPOT is then used as the ground range between the points, but a magnetic bearing of the roll-in point reference the track point is still required. This is found by

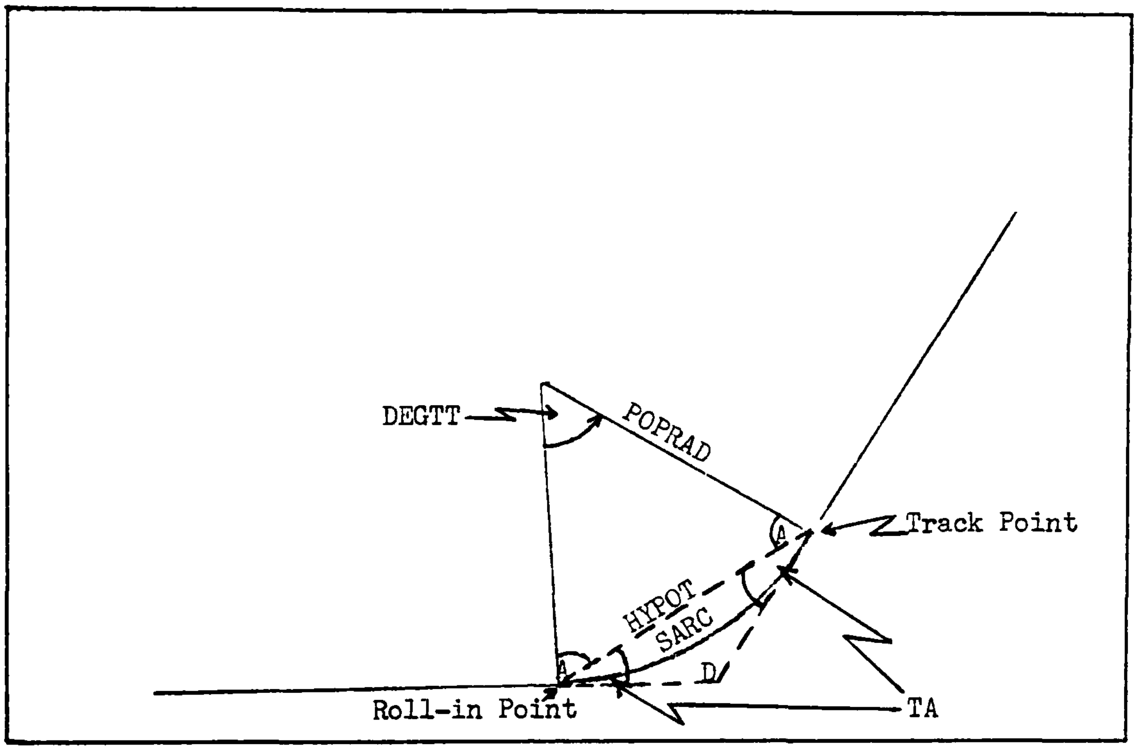


Fig. 12. Geometry for Roll-in Point to Track Point

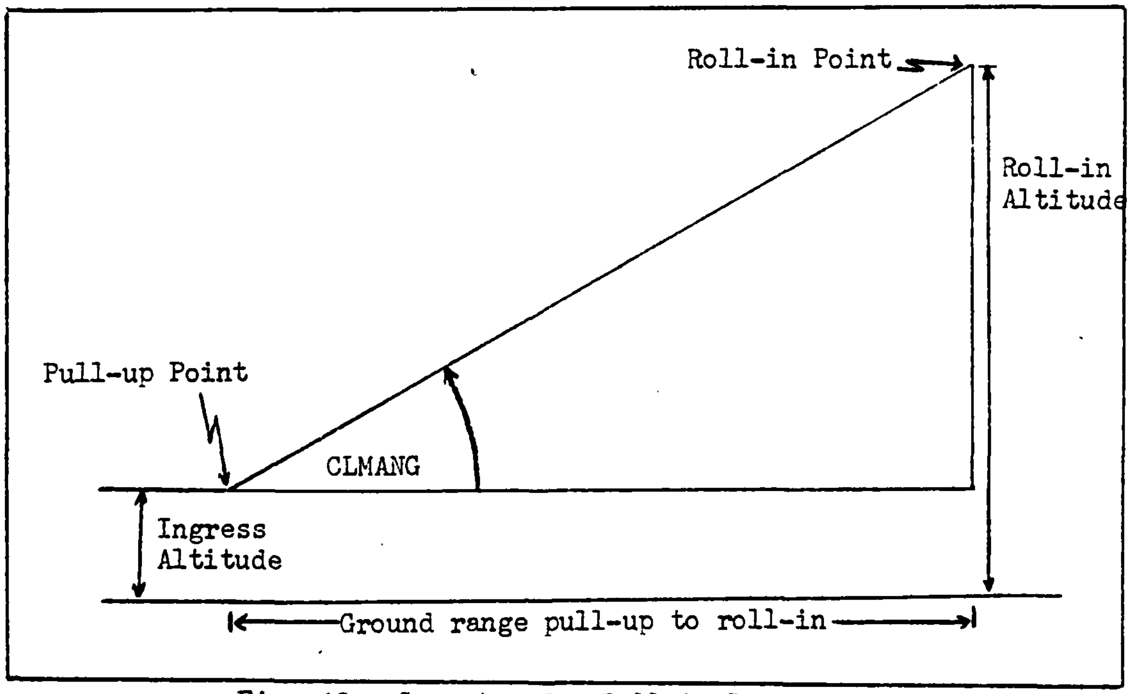


Fig. 13. Geometry for Roll-in Point Altitude

obtaining an angle TA:

$$TA = (180 - D7)/2.0 \qquad (38)$$

where D7 = 360 - 180 - DEGTT
DEGTT = degrees to turn

This angle TA is then added to the run-in heading if the roll-in direction is left, or it is subtracted from the run-in heading if the roll-in direction is right. The required magnetic bearing then becomes this temporary heading minus 180 degrees. The bearing and ground range are then used to obtain the coordinates for the roll-in point.

The pitch angle at the roll-in point equals the planned climb angle, CLMANG, where:

$$CLMANG = ADIV7 + 5.0 \qquad (39)$$

where ADIV7 = aircraft dive angle (degrees)

The altitude of the aircraft at the roll-in point can be determined by the equation:

$$RIALT = APXALT - (60)(CLMANG) \qquad (40)$$

where RIALT = roll-in altitude (feet)
APXALT = apex altitude during turn to track point

If the planned dive angle is 10 degrees, the apex altitude is:

APXALT = ACFT(i,12) + 1000.0

where ACFT(i,12) = release altitude of first weapon, feet.

If the planned dive angle is 15 degrees, the apex altitude is:

APXALT = (2)((ADIV7)(100.) + ACFT(i,12)/(2.0) (41)

where ADIV7 = planned release dive angle (degrees)

The final requirement for the roll-in point, the heading at roll-in, is found by calculating the ingress heading. For a left roll-in to final heading, ingress heading equals run-in heading plus 30 degrees. If the roll-in to final is right, the ingress heading equals run-in heading minus 30 degrees.

The next point, the pull up point uses the ingress heading run-in 180 degrees as the magnetic bearing. The relationships in Figure 13 are used to calculate the ground range back to the pull-up point. The ground range, GR, is, therefore, found by the equation:

GR = (Sin((90-CLMANG)/57.3))(ACFT(i,38)-ACTIN7)/Sin (CLMANG/57.3) (42)

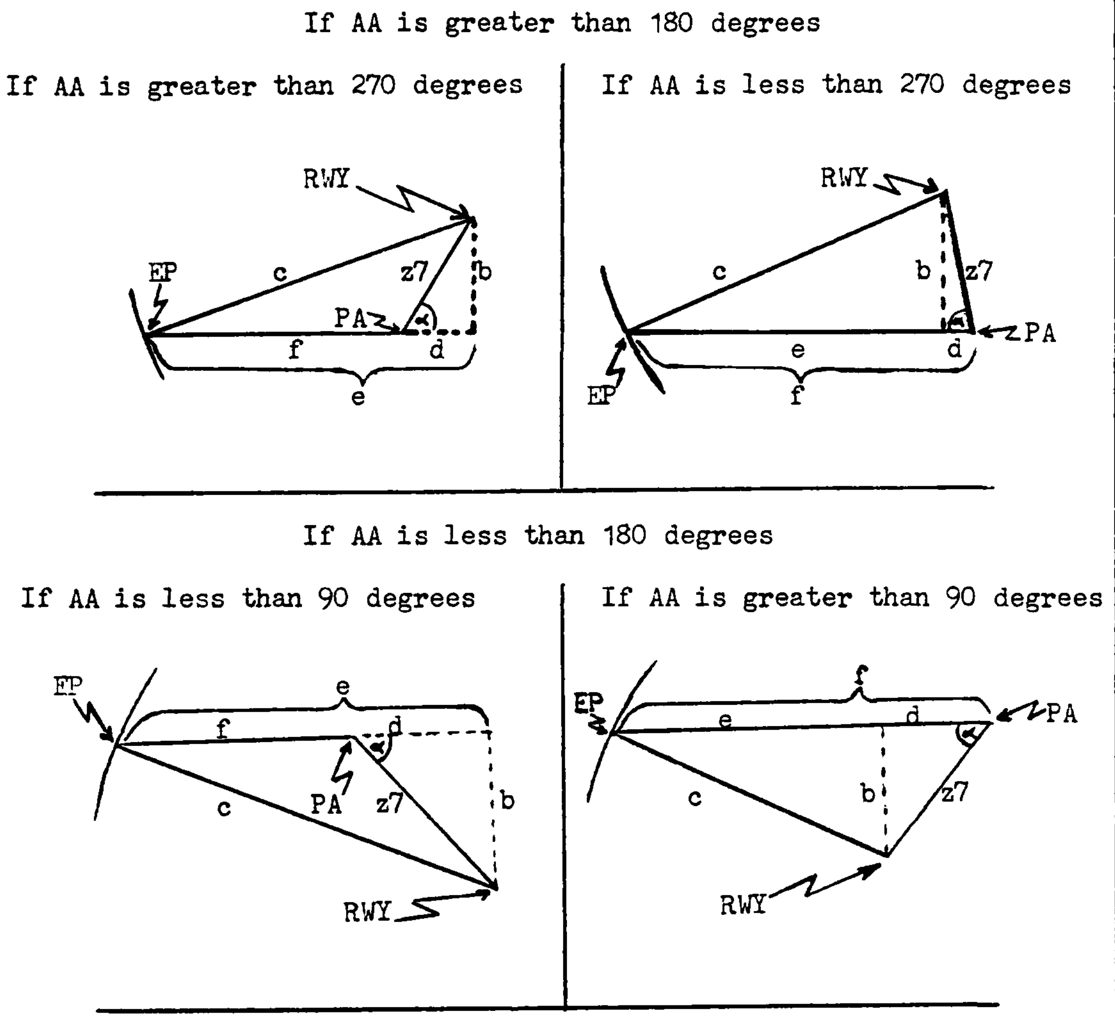
where CLMANG = aircraft climb angle (degrees)
ACFT(i,38) = aircraft altitude at roll-in point
ACTIN7 = ingress altitude (feet)

This ground range and magnetic bearing is then put into subroutine XYCOR1 to obtain the coordinates of the pull-up point. The desired pitch and heading into the pull-up point are 0 degrees and the runin heading respectively.

The calculation of the entry point to the target area requires logic based on the geometric relationship depicted in Figure 14. The reason for this is the exact ground range of the pull-up point to the 10 mile target perimeter is unknown. The coordinates of the pull-up point and runway center are entered into subroutine MBRAN2 to obtain the relationship of the pull-up point to the runway center. The ground range F between the pull-up point and unknown entry point is then calculated. This range and magnetic bearing are then used with subroutine XYCOR1 to find the coordinates of the entry point. The desired altitude, pitch angle, and heading at the entry point are the ingress altitude, 0 degrees, and ingress heading respectively.

If, after calculation the track point earlier in the program, it is noted that the delivery is going to be level rather than a pop-to-angular, the information for the level off, climb point, and entry point are calculated with logic similar to that described for the pop-to-angular. The level delivery logic is simpler, however, in that no turn is made to complicate the planning as does the roll-in to track point calculations for the pop-to-angular delivery

Finally, after the information is calculated for each


If AA is greater than 180 degrees
If AA is greater than 270 degrees
RWY
EP
c
z7
b
PA
α
f
d
e
If AA is less than 270 degrees
RWY
c
b
z7
α
PA
e
d
f
EP
If AA is less than 180 degrees
If AA is less than 90 degrees
EP
e
f
d
α
PA
c
z7
b
RWY
If AA is greater than 90 degrees
EP
f
e
d
PA
α
c
b
z7
RWY

where EP = Entry point
PA = Point after EP (climb point or pull-up point)
RWY = Runway center
AA = Aspect of RWY from aircraft located at PA
z7 = Ground range from RWY to PA
b = Radius from RWY to aircraft flight path
c = Ground range from RWY to intersection of aircraft flight path and target area perimeter
d = Ground distance from PA to intersection of b and flight path
e = Ground distance from EP to intersection of b and flight path
α = Acute angle formed by flight path and RWY
f = Ground distance back to EP from PA

Fig. 14. Entry Point Geometry

aircraft's navigation points, a total distance is calculated and an estimated time enroute (ETE) is determined for the aircraft based on aircraft velocity. This ETE is stored in array ACFT and used later to schedule the times the aircraft enter the area to insure against fratricide due to bomb fragmentation over the target. In addition, all the navigational information, the x, y, z coordinates, the desired pitch and desired headings are stored in array ACFT.

Subroutine IMPCT8

The purpose of subroutine IMPCT8 is to calculate the x, y, z coordinates of the current predicted collision course impact point of a SAM and its target aircraft. The output of the subroutine also includes the missile heading and pitch angle which the missile must fly to reach the impact point. The heading and pitch calculations do not address turn radius. This is compensated for by calling the subroutine frequently during the missile's time of flight. The slant range and time to impact are also outputs of this routine.

The logic for this subroutine differs from the impact logic described by Neal and Kizer (Ref 16:74-77). In their research, the impact point calculations were based on a ratio of aircraft velocity to missile velocity. While this is satisfactory for a two-dimensional case in which the plane of intercept is parallel to the ground, it is not satisfactory for an intercept plane that is not parallel to the ground. As a result, logic was developed for subroutine IMPCT8 which does permit calculation of a three-dimensional intercept problem regardless of the orientation of the intercept plane with respect to the ground. The logic for the subroutine is based on concepts explained by Dr. Charles Bridgeman (Ref 5).

The inputs for the subroutine are the current x, y, z coordinates of the aircraft and missile in the runway centered coordinate system. Also needed are the rates at which the aircraft changes its x coordinate, y coordinate, and z coordinate. The reference system of the aircraft and missile is then temporarily set in a missile centered coordinate system. As depicted in Figure 15 the missile's current position is the center of this system whose xy plane is parallel to the actual ground. The coordinates of the aircraft in this missile centered system are represented by PZXRF, PRYRF, and PZZRF. The length of the vector SRΦ is initially calculated and designated as S1. The time for the missile to fly the distance S1 is computed and called TTI8. The changes in aircraft x, y, and z position during this time TTI8 are then computed and added to PZXRF, PZYRF, and PZZRF to get the coordinates of where the aircraft will be after time TTI8. A vector SR1 is drawn from the missile's current position to where the aircraft will be TTI8 seconds in the future. The magnitude, S2, of vector SR1 is then compared to the original value of S1. If the difference between S2 and S1 is less than five feet, then the aircraft position at the end of time TTI8 is where impact will occur if the aircraft does not maneuver. However, if S2 and S1 are more than five

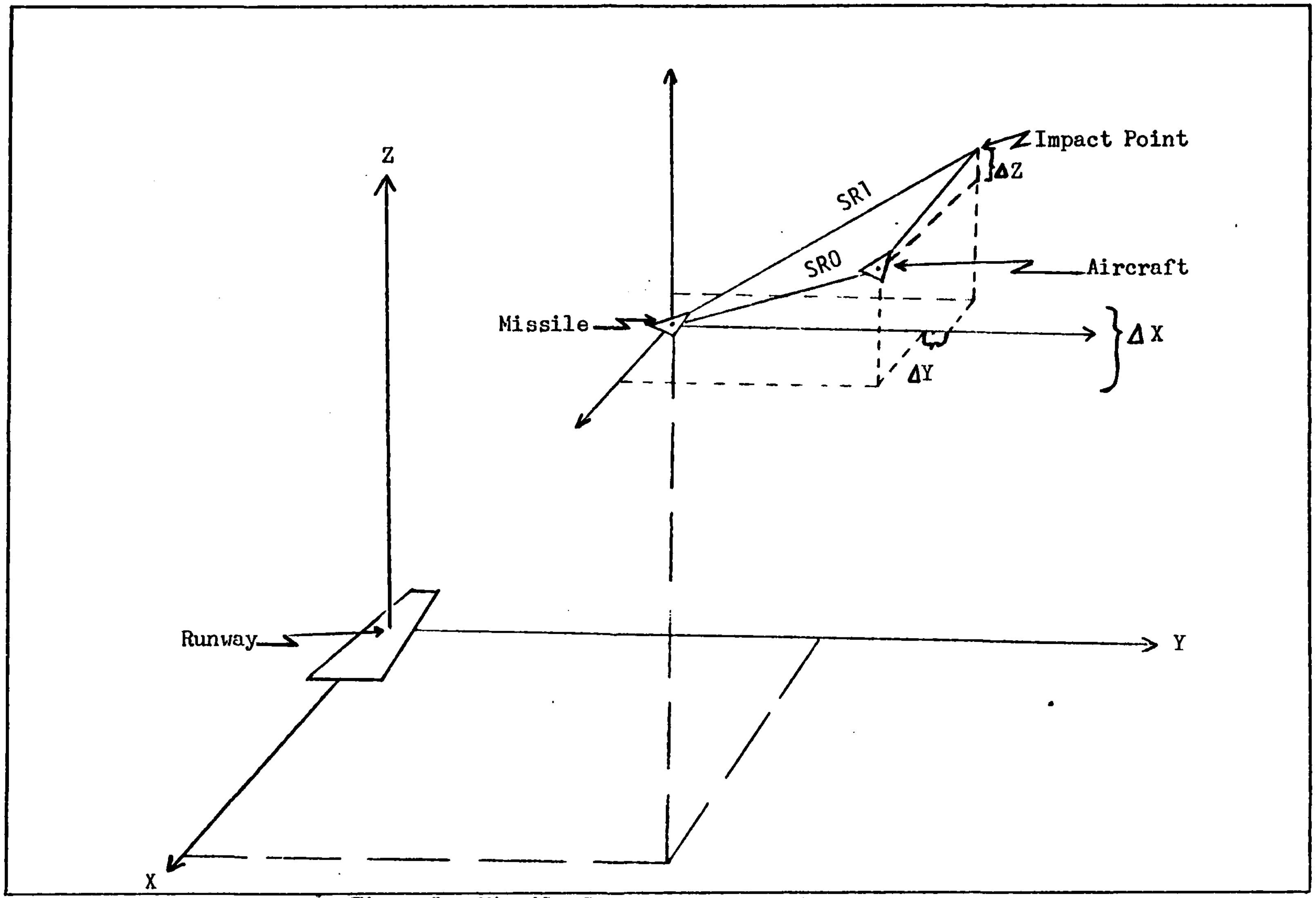


Fig. 15. Missile Centered Coordinate System

feet apart, an interative process is used to determine the final impact point by repeating the calculations. If the aircraft is moving away from the missile the value of S1 is reset to that of S2. If the aircraft is moving toward the missile, the S1 value is reset to an average of S1 and S2 values prior to the next iteration.

The subroutine then calculates the impact point coordinates in terms of the runway centered coordinate system. The missile heading required to arrive at the impact point is determined through the use of MBRAN2 and the missile pitch is calculated by comparing the missile's height to the planned impact height.

It is possible that the planned impact point could lie below the 100 foot minimum SAM engagement altitude. One time this could occur is when the aircraft descends below 100 feet during egress from the target area. The subroutine will detect this and redefine the z coordinate of the impact point to be 100 feet above the ground. The result in this case will be a missile detonation above the aircraft during egress. This type of situation is depicted in Figure 16.

Finally, the subroutine compares the maximum time of flight remaining for the missile to the required time to impact. If the time required to impact exceeds the maximum time of flight remaining for the missile, then the subroutine indicates that the missile will not catch the aircraft given the aircraft's current flight vector.

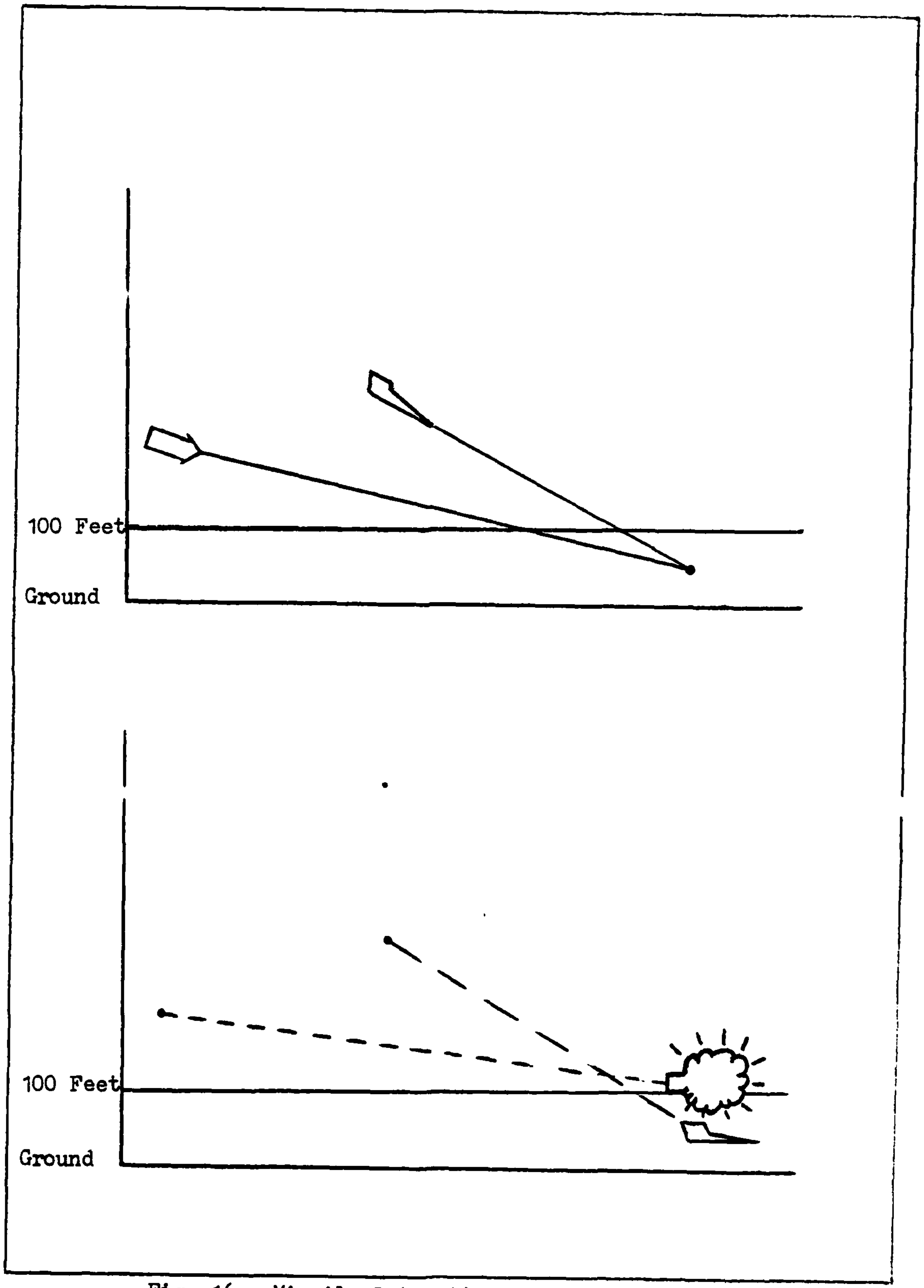


Fig. 16. Missile Detonation at Minimum Altitude

Subroutine BREAK3

The purpose of subroutine BREAK3 is to provide a direction of turn, a heading, and a maximum turn capability to the pilot who has decided to perform a SAM break maneuver against a missile. The entering arguments are the missile's identification number, the missile's target aircraft, and the coordinates of the aircraft and the missile. Subroutine MBRAN2 is called to determine the aspect of the missile to the aircraft. Subroutine BREAK3 then assigns a maximum performance 8 G turn toward the missile in an attempt to reduce the missile's ability to compensate for such a turn. In addition, the subroutine provides a desired heading for the aircraft.

Subroutine PKAA10

Subroutine PKAA10 assigns a probability of an aircraft kill due to AAA. As described in Chapter II, the AAA threat is a one second burst. The entering arguments are the slant range, SR10, from the aircraft to the AAA site; the average vulnerable area, AV10, of the aircraft; and the aircraft G loading, TGTG10.

Subroutine PKSAM9

An aircraft's probability of being killed by a missile is calculated in subroutine PKSAM9. Although the initial probability of kill (PK) is assessed at missile launch, the missile's inflight Pk can vary due to changing engagement conditions. As the aircraft maneuvers, the location of the predicted impact point varies. As this planned impact point changes, so does the corresponding range of that point to the threat site. Since CEP depends on the value of this range, the CEP and resulting PK will also vary.

To compensate for variations in engagement conditions, subroutine PKSAM9 is called at one second intervals to update the value of the anticipated Pk which corresponds to the current planned impact point. The subroutine is also called at the missile detonation time to calculate the actual Pk. If the subroutine is called due to a missile impact location update, the subroutine uses an equation used in previous works (Ref 16:60). This equation, the rationale for which is described in Chapter II, is:

$$Pk = 1.0 - .5^{(LR/CEP)^2} \tag{12}$$

where Pk = probability of killing the aircraft
LR = lethal radius of the missile warhead
CEP = circular error probable of the missile's guidance system

This equation is not appropriate, however, when the aircraft actively maneuvers against the missile. An assumption of the equation is that the aircraft is at the predicted impact point. Miss distance against the nonmaneuvering target, therefore, is merely a function of missile guidance error as calculated in the equation. If, however, the aircraft maneuvers at a time and in a manner such that the missile can not adequately update its impact point prior to detonation, the aircraft will not be at the impact point at detonation. In this case the miss distance is a function of both missile guidance error and slant range of aircraft from detonation point. It is for this reason that the cell Pk solution is used.

The cell Pk solution, as described in Chapter II, divides the plane of impact into 109 cells. The 109 cell method is used due to its significantly increased accuracy over the 10 cell method. This plane of impact is defined by the plane through the planned impact point that contains the flight vector of the missile and the flight vector of the aircraft used in determining the impact point. The aircraft's slant range from each of these cells is computed at detonation time, compared to the lethal radius, and a probabiliy of kill is assigned. This cell Pk solution is used when the subroutine is called due to a missile detonation since it accounts for both a maneuvering and nonmaneuvering target. The Pk equation is used at all other times as it is satisfactory for the nonmaneuvering target and minimizes computation time.

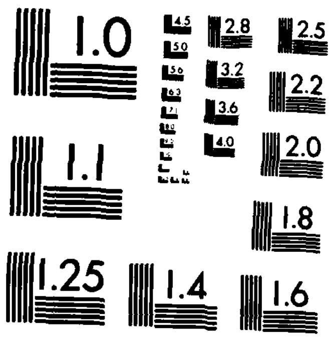
Subroutine EVENT

Subroutine EVENT is the most important and lenthy subroutine in this program. The subroutine is used to move from the scheduled SLAM events logic to the actual events themselves. It is not one event, therefore, but a large assortment of events which are triggered by SLAM state variables passing specified values. Some of these values are constants, but others are global variables which are revalued many times within the program.

The first two events of the 34 events in the subroutine are used to give each aircraft an initial velocity. Once initialized each aircraft maintains this airspeed throughout the scenario.

Event 3 is used to provide aircraft 1 navigational information. The event is called as the aircraft passes a navigation point and needs information about the next navigation point. This information, the next coordinate's x, y, and z coordinates as well as the necessary heading and pitch to reach that point are stored in global values [xx(15-19)]. The x, y, and z coordinates are usually what have been stored in array ACFT by the flight planning subroutine ROUTE7. The heading to the next point [xx(18)] is found via a call to subroutine MBRAN2. The pitch angle required to get to the next navigation point is computed by comparing the aircraft's current altitude, the desired altitude at the next point, and the range to the next point. This range to the next navigation point is then stored as a state variable [ss(22)]. A special case occurs in the pop-to-angular delivery. To insure the aircraft rolls out close to the flight planned target attack heading, the desired heading into the turn point is the heading into the

AD-A141 324   A SIMULATION MODEL TO EVALUATE AIRCRAFT SURVIVABILITY   2/3
AND TARGET DAMAGE D..(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI..
UNCLASSIFIED   M J FOLEY ET AL. MAR 84 AFIT/GST/OS/84M-10   F/G 9/2   NL


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

track point obtained from the flight planning routine ROUTE7 and stored in array ACFT.

Once the desired navigational information is obtained, a check is then made to see if the desired heading and pitch vary from aircraft number one's current heading [ss(4)] and pitch [ss(5)]. If there are differences, a heading flag and/or pitch flag is set to indicate an acceleration in heading or pitch is required. In addition to setting these flags, a direction is set (right or left for heading and up or down for pitch). The aircraft G loading during navigational turns is 4G. A similar event, event 4 is used to perform these same types of calculations for aircraft 2. In the case of aircraft 2, however, the data for the next navigation point is stored in different global variables [xx(25-29)].

It is also in events 3 and 4 that the probability of aircraft arrival at the last weapons release point and probability of target damage are assessed for each aircraft. The probability of arrival is determined by the equation:

$$PA = (1-PK1)(1-PK2)(1-PK3)(1-PK4)(1-PK5)(1-PK6) \quad (43)$$

where PA = probability that the aircraft survives through weapons release.
PKi = current probability of kill assessed against the aircraft from each of the four missiles and two AAA sites.

The probability of target damage is then determined by the relationship:

$$PTGTDMG = (PA)(PDMG)(PSEE) \tag{44}$$

where PTGTDMG	=	probability of target damage by the aircraft.
PA	=	probability pilot arrives at target.
PDMG	=	probability of damage given pilot bombs the target (JMEM calculations).
PSEE	=	probability that pilot can see target

Events 5, 6, 7, 8 are used to reset the heading and pitch flags of the aircraft to zero. Events 5 and 6 reset the heading flags of aircraft 1 and 2 respectively and rest the G loading to 1.0. Events 7 and 8 are used to reset the pitch flags of aircraft 1 and 2 respectively. The effect of setting a flag to zero is to cause the aircraft to maintain the current heading or pitch.

Events 9 and 10 are used to direct aircraft 1 and 2 respectively out of the target area. These events may be called after an aircraft completes its 45 degree turn after weapons release, or may be called after an aircraft has completed a "SAM break" against a missile. The event uses the aircraft's current position and the coordinates of the center of the airfield runway to obtain the aspect of the airfield to the aircraft via subroutine MBRAN2. The aircraft is then programmed to turn in the shortest direction to a heading that equals this bearing and exit the target area on this heading. The aircraaft is also directed to begin a descent to 50 feet altitude to get below the minimum SAM engagement coverage.

Events 11 and 12 are used to schedule abrupt level-offs, that is set pitch and pitch rate to zero, for each aircraft as the aircraft arrives at 50 feet. Events 13 and 14 are scheduled when aircraft 1 and 2 respectively reach the 10 mile radius circle and, therefore, exit the target area. The aircraft velocity is set to zero and the status is set to idle. It is also in event 14 that the probability of survival for each aircraft is calculated as well as the overall mission survival rate and overall target damage as a result of the activities of both aircraft.

Events 15 and 16 are used to trigger the SAM engagement logic for each aircraft respectively. The event is initially scheduled when an aircraft climbs above 100 feet anywhere within the 10 nautical mile radius circle. A check is made to see which of the SAM sites are ready to begin acquisition and tracking of the target. The requirements for such a ready status are the site not currently tracking a target, confounding delay complete, and one or more missiles available to launch. For each site that is ready to begin acquisition and tracking of the target aircraft, an acquisition and tracking completion time or ready-to-fire times is scheduled. This ready-to-fire time depends upon the value of the acquisition and tracking requirement for each specific site.

When the current time reaches the ready to fire time for a site, event 17 or 18 is called. Event 17 is called for SAM site 1, and event 18 is called if current time reaches the ready-to-fire time for SAM site 2. Initially, in each event, the site is merely identified as 1 or 2 a.d variable ITHRT set equal to it. Then, the logic proceeds to common launch logic where the missile launch decision begins. For this decision, the SAM site's coordinates, the aircraft coordinates, the aircraft's x, y, and z velocity components, and velocity of the missiles from that site are inputted into subroutine IMPCT8. If subroutine IMPCT8 indicates that the projected impact point for an immediate launch lies beyond maximum missile range or inside minimum range then a delay of five seconds is scheduled prior to the site once again deciding whether or not to engage. If the target aircraft is within missile minimum and maximum range, a missile is scheduled to be fired one second later from that site. If the missile to be fired is the site's first missile, a ready-to-fire time based on a ten second delay is scheduled for the second missile. In addition, a check is made to see if the other missile site is tracking the same aircraft for a possible engagement. If it is, then the status of the other site is changed to zero. If the missile scheduled for firing is the second missile to be fired by the site, the site status is changed to zero due to a missile reloading requirment.

Events 19 and 20 are used to assign threat sites 3 and 4, the AAA sites, against an aircraft. Either of the events is called when an aircraft is within the AAA site's maximum range and the tracking delay has elapsed since the aircraft entered the AAA engagement ring. The average target vulnerability is assigned and subroutine PKAA10 is called to assign a probability of kill by that AAA site against the aircraft. Events 21 and 22 are used to disengage an AAA site from an aircraft when the aircraft range from the AAA site exceeds the maximum range of the AAA weapon.

Events 23, 24, 25, and 26 are missile launch and predicted impact point update calculations for missiles 1, 2, 3, and 4 respectively. If one of these events is scheduled, a check is made to see if the missile's launch time (global variables xx(71) through xx(74)) is nonzero. A zero value indicates the missile is not yet airborne, but a nonzero value is an airborne missile's last time that the missile's planned impact point was updated. If the missile is not yet airborne, then it is launched. The number of missiles available at the site is reduced by one, the missile's target aircraft is assigned from the site, and a maximum time of missile flight is programmed via a missile destruct time that equals launch time plus maximum time of flight remaining. The launch time is also assigned to array MSL.

As a missile is launched, it uses the same logic as airborne missiles to update the predicted impact point information. This logic begins by calculation of the spherical coordinate system angles θ and Φ which represent the missile's heading and pitch since last update of missile impact point. This is similar to the relationships between aircraft heading and pitch and the spherical coordinate systems angles θ and Φ portrayed in Figure 9. The missile's current x, y, and z coordinates are then calculated as a function of its position at the last calculation of predicted impact point, the time since that last update and the change in missile x, y, and z due to its flight vector. The missile's current coordinates, velocity components, aircraft's coordinates, and the aircraft's x, y, and z components of velocity are then inputted into subroutine IMPCT8 in a similar fashion as described for events 17 and 18.

This logic for events 23, 24, 25, and 26 contain an addition which events 17 and 18 do not. In events 23, 24, 25, and 26 a probability of awareness is assigned. This is the probability that the pilot will be aware of the missile. If this number indicates that the pilot will be aware of the missile, then the AWARE is set to indicate this. (This is used later as the time to impact approaches zero.) Then prior to calling subroutine IMPCT8, the maximum remaining flight time of the missile is calculated. The subroutine IMPCT8 returns the planned impact point coordinates for current missile and aircraft flight vectors. It also provides the pitch and heading which the missile must fly to reach this impact point and the slant range and time to that impact point.

The subroutine PKSAM9 is then used to assess the SAM PK against the aircraft given the revisd impact point. The slant range from the threat site to the impact point is calculated via subroutine MBRAN2. The aircraft slant range to the impact point is also calculated for use at time of warhead detonation in the cell Pk solution. A jamming option is set to represent either a jamming or nonjamming aircraft prior to a call to subroutine PKSAM9 to obtain the Pk. A check is then made to see if the Pk of the missile exceeds the Pk, which, if perceived by the pilot, will cause the pilot to jettison his weapons and actively maneuver against the SAM. If the missile Pk is less than this value, there is no action taken by the pilot, the pilot continues his planned route profile, and the next impact point update is made in one second. If, however, the Pk of the missile exceeds the value at which a pilot would react against the SAM, then a check is made of time to impact and pilot awareness. If the time to impact exceeds two seconds, or if the pilot is unaware of the missile's presence, then no defensive action is taken by the pilot. Otherwise, subroutine BREAK3 is called. At this time the aircraft flag for a defensive maneuver [ACFT(i,11)] is set to 1.0, and the impact point is no longer updated. This indicates the inability of the missile with relatively short time to impact to be able to effectively move its planned impact point in reaction to an aggressive aircraft defensive maneuver.

Events 27, 28, 29, and 30 are the logic for occurrences at the actual detonation times for each of the missiles. These events identify the missile that is detonating. For each detonating missile, the range from the impact point to launch site is computed and a CEP determined for the time of impact. The cell Pk solution is used to assess Pk at detonation time, whereas the equation solution of Pk is used for all planned impact point update calculations. After a SAM Pk is assessed against the aircraft, event 9 or 10 is called to route the aircraft out of the target area. The SAM PK is then stored in array ACFT.

Finally, events 31, 32, 33, and 34 are called to schedule AAA fire against the aircraft. Events 31 and 32 represent calculations for AAA site 1 against aicraft 1 and 2 respectively. Events 33 and 34 represent calculations for AAA site 2 against both aircraft. In each case, an AAA tracking completion time is scheduled. It is at this completion time that the AAA can fire at the aircraft and the previously described AAA Pk calculations are made. As with the SAM PK, the AAA PK is stored in array ACFT.

Summary

The aircraft survival and target damage questions posed by this research involve concepts which are adequately represented by a discrete-continuous simulation model. The SLAM language is used, but minimum dependence is placed on it. The continuous capabilities of SLAM are incorporated, but standard Fortran logic is used extensively. The model includes an analytical approximation of a JMEM hand-calculation method to compute weapons effects, linear approximations of a graph from JMEM to assess the probability of seeing the target, and an extensive threat and pilot reaction scenario to determine the actual probability of the pilot to arrive at the target and to successfully exit the target area. A computer listing of this model is contained in Appendix B.

IV. Verification And Validation

The credibility of a model depends in large part on its verification and validation. These two processes are performed to insure the model performs as expected and to see if its output is representative of the system it models.

Verification

The verification process is the determination of whether or not the model performed as intended. Verification is not a determination of whether or not the model's output is characteristic of the system being investigated. Rather, verification is used to investigate the suitability of the model's output given the input and the model's processes.

This research used five basic verification techniques described by Law and Kelton (Ref 13:334-336). First, the model was written in modules and transformed into subprograms. These subprograms were verified prior to incorporation into the general model. Second, a structural walk-through technique was used. After writing the code for a subroutine, an analyst "walked" the other analyst through the code to check for logic or syntax errors. Third, numerous PRINT statements and a trace were used. These PRINT statements were placed to identify system changes following the occurence of events in the model. The trace was used to show the continuous movement of the aircraft with respect to the navigation points and threats. Fourth the model was run and the results compared with hand calculated results. Finally, the movements of the aircraft and missiles were analyzed graphically.

The model contains the nine major subroutines and one Fortran function described in chapter III. The model also contains a major subroutine, subroutine EVENT. The following is a description of the verification of these components of the model.

Subroutine XYCOR1. The desired outputs of subroutine XYCOR1 are the coordinates of a point B given that point's relationship to another point A whose coordinates are known. The subroutine was verified by positioning point A in each of the four cartesian quadrants, on each of the x, y axes, and at the center of the coordinate system. In each of these cases the magnetic bearing and range to a point B were inputted into the subroutine. Further, for each case, the position of point B was rotated through 360 degrees around point A at 15 degree increments. The computer derived coordinates were then compared to hand calculations and graphpaper plots.

Subroutine MBRAN2. Subroutine MBRAN2 was developed to determine the magnetic bearing, ground range, slant range, and aspect angle of a point B reference to a point A given the coordinates of each point. Verification of this subroutine involved three phases. The first phase was to rotate the heading of the entity at point A through 360 degrees at 15 degree increments and note that aspect angle alone changed. The second phase was to vary the altitude of the point B, hold the x and y coordinates of A and B constant, and note changes in slant range. Finally, the location of point B was moved about in the different quadrants of the xy plane. Comparison with hand calculations indicated satisfactory program output.

Subroutine SCAN4. Subroutine SCAN4 is used to assess threat locations reference to the target and to produce optimum threat avoidance run-in headings and turns to final run-in headings for each aircraft. The subroutine incorporates subroutines XYCOR1 and MBRAN2. The subroutine was verified by placing different numbers of threats in each of the two possible approach cones to the target. The subroutine consistently produced the correct information as verified by plots on graph paper.

Function AREA5. Function AREA5 is a numerical integration routine used by subroutine JMEM6 to calculate fractional coverage of the target area. Prior to incorporation into JMEM6, this Fortran function was verified against hand calculated results. The results from the function compared favorably with the hand calculated results as indicated below:

Hand Calculation	Function AREA5
.8862	.8862
.8856	.8855
.8793	.8862
.4431	.4426

Computer results were also compared to results obtained from a graph used by the JMEM logic. Once again, the computer values compared favorably:

JMEM Graphical Solution	Function AREA5
.59	.59
.98	.98
.79	.79
.97	.96

Subroutine JMEM6. Subroutine JMEM6 was developed to perform the manual's analytical calculations of target damage due to weapons effects. As noted in Chapter III, this subroutine makes use of approximations of some table and graphical data from JMEM, approximations which are adequate for the scope of this research. The most critical output of the subroutine is the target damage, but several other pieces of information, including ground range of release point to the target, are used by other parts of the program. Finally, the probability of seeing the target, though not a part of the analytical method in the manual, was incorporated into the subroutine.

In order to verify the computer program's probability of target damage, four different scenarios were hand calculated using the time consuming JMEM methodology. The scenarios were chosen to represent all the possible tactics of this research. As a result, the verification scenarios consisted of a level attack with high drag weapons, a level attack with low drag weapons, a pop-to-low angle delivery with high drag weapons, and a pop-to-low angle delivery with low drag weapons. The parameters of each scenario, including the release altitudes and the intervolometer settings, were characteristic of the specific scenario. A comparison of the resulting probabilities of damage due to weapons effects are as follows:

	JMEM Hand Calculation	Computer Program
Level, high drag	.026	.026
Angular, high drag	.051	.052
Level, low drag	.033	.033
Angular, high drag	.060	.059

The hand calculated ground ranges and slant ranges also compared favorably with the computer program product.

The computer program used linear approximations of a JMEM graph to produce probability of the pilot seeing the target at the release point (Ref 20:C-13). Results using the program logic were compared to results using the JMEM graph. For altitude regimes between 0 feet and 2500 feet and for ground ranges of 2000 to 7000 feet, the greatest error was only .06, but the typical error between the graph and program was .02 or less.

Subroutine ROUTE7. The purpose of subroutine ROUTE7 is to plan an ingress route of flight to the target. The method used in the subroutine is to start at the target and plan back to the entry point for each aircraft. The subroutine uses bomb range information from subroutine JMEM6, roll-in directions and run-in headings from subroutine SCAN4, and geometric relationships from subroutines XYCOR1 and MBRAN2. As these other subroutines were verified prior to the development of subroutine ROUTE7, no problems arose during its verification involving several different scenarios. In each case the computer program derived coordinates agreed closely with hand calculated and graphical results. The results of one of these scenarios, a high drag level delivery by aircraft number one and a high drag pop-to-angular delivery by aircraft number two, are contained in Table IV. The target information and threat locations for this scenario were as defined in the scope of this research.

<u>Subroutine IMPCT8</u>. Subroutine IMPCT8 was designed to provide the time to impact and impact point coordinates given the aircraft and missile locations and flight vectors. The allowable error in the calculation of the impact point is five feet. The subroutine was verified prior to incorporation into the general program by investigating several scenarios. In each scenario the missile position after the program's elapsed time-to-impact was compared to the aircraft's position. The first scenario was an aircraft directly above the missile and flying straight up. The second was an aircraft displaced 20,000 feet east from the missile and flying directly north. The third was similar to scenario two but allowed the missile a maximum time of flight

TABLE IV

Results of Verification of Coordinates

Aircraft #1	HAND CALCULATED COORDINATES IN FEET		COMPUTER OUTPUT COORDINATES IN FEET	
	x	y	x	y
Last Weapon Release Point	2588	-4987	2595	4983
First Weapon Release Point	1903	-5467	1909	-5463
Track Point	-1729	-8009	-1721	-8005
Level Off Point	-3909	-9536	-3900	-9531
Climb Point	-5080	-10355	-5070	-10350
Entry Point	*	*	-45743	-38827

Aircraft #2				
Last Weapon Release Point	2976	-5462	2975	-5463
First Weapon Release Point	2503	-6138	2502	-6139
Track Point	-1	-9714	-1	-9713
Roll-in Point	-2424	-11748	-2421	-11744
Pull-up Point	-5029	-12962	-5028	-12959
Entry Point	*	*	-49610	-33746

*Note: The entry point coordinates were verified by comparing computer values of each entry point with the ten nautical mile radius circle and the bearing from the next point after entry point. In each case the computer generated entry point was exactly 60,000 feet from the runway center and at the desired bearing from the navigation point following the entry point.

of two seconds. Finally, in scenario four, the aircraft was located 20,000 feet away from the missile with a flight vector pointed directly at the missile. Hand calculations of cases one, two, and four confirmed the computer results. That is, the missile and aircraft were within five feet of each other after a time equal to the program's output of time-to-impact. In case three, the subroutine correctly predicted that the impact point was beyond the missile's maximum time of flight. Finally, the subroutine correctly assigned a minimum impact altitude of 100 feet above the ground when the aircraft was below 100 feet.

Subroutine PKSAM9. Subroutine PKSAM9 was developed to determine the probability of a missile kill of either a maneuvering or nonmaneuvering aircraft. As described in Chapter III, an equation is used to calculate the Pk for planned impact point positions. At the detonation time, however, a cell method is used to calculate the Pk. These cell Pk calculations were verified in three steps.

First, a 10 cell model was built and an aircraft placed in the center cell. This represented the case of a nonmaneuvering target. With this scenario, hand calculations of Pk were made for both jamming and nonjamming aircraft. The results were compared to Pk calculations using the equation form. The results were favorable but greater accuracy was needed than was available in the 10 cell model. This resulted in the second step, the development of a 109 cell model.

Once again an aircraft was positioned at the center of the model and the results of both jamming and nonjamming scenarios were compared to the equation's results. The results of the cell method were identical to the equation method. During this second step, the aircraft was also placed at positions other than the exact center of the model. Hand calculations of the slant ranges to various cells verified that the computer output was correct.

The third step involved the verification of the subroutine after it was incorporated in the main model. The slant ranges from the aircraft position to the center of cells one through ten were hand calculated. These results confirmed the accuracy of the computer output. In addition, two cells from each of the models remaining rings were chosen and slant ranges to these cells calculated. Once again, the hand calculations confirmed the accuracy of the computer output.

Subroutine PKAA10. The purpose of subroutine PKAA10 is to calculate the aircraft's probability of kill when fired upon by AAA. The subroutine was verified in two steps. First, ten different slant ranges and aircraft G loadings were put into the program. The resulting values of PK were identical to hand calculated results. The second step was the verification of the subroutine's output after the subroutine was incorporated into the main model. Print statements were used to obtain the computer calculated values of PK. These values were verified correct via hand calculated results.

Subroutine BREAK3. Subroutine BREAK3 was developed to provide a turn direction, a heading, and a maximum turn rate acceleration capability to the pilot who performs a defensive maneuver against a missile. Prior to incorporation into the main program, ten different scenarios were investigated. In each scenario the missile was placed at a different aspect to the aircraft. Hand calculations and graph paper plots confirmed that in each scenario the scheduled turn direction and required heading were correct. Further, in each case, the subroutine scheduled an 8G lateral acceleration.

Subroutine EVENT. In addition to the verification of the individual subroutines described above, the interaction of subroutine EVENT with the other parts of the program was verified. As noted in chapter III, the purpose of this important subroutine is to permit scheduled events to occur, events which occur as a function of time and/or position of the entities in the target area. The major functions of the subroutine can be classified in one of two categories. The first of these is the three-dimensional routing of the two aircraft from navigation point to navigation point, the accomplishment of the desired weapons deliveries, and finally the exits from the target area. The second major task is the execution of the threat logic, that is the actions taken by the threat sites and their weapons against the aircraft. Included in this latter task are the reactions of the pilots to the threats.

In order to verify the models performance of each of these two major tasks, the verification was performed in two phases. The first phase verified the performance of subroutine EVENT in routing the aircraft through the target area. This verification is contained in Appendix C. The second phase, the verification of the threat logic and pilot reactions to the threats is incorporated in Appendix D.

It should be noted that during the verification of the routing and threat logic of subroutine EVENT, several significant observations were made. These included limitations of the SLAM SEVNT input statement logic, required tolerances in determining navigation point passage, and the effect of SAM Pk on the nonjamming pilot's decision to break against the SAM.

In SLAM, the SEVNT input statement is used to call a specific event when some SLAM variable passes a defined value (threshhold). The original program for this research contained 40 such SEVNT statements. It was then discovered that a maximum of 25 such input statements can be used in the SLAM language. As a result, 25 of the most critical threshhold occurrences, situations which repeatedly occur during each computer run, were left in SEVNT input statement logic form. The remaining occurrences, most of which occur only once during each computer run, were either incorporated into subroutine STATE with Fortran "if-then" logic, or were scheduled via the SCHDL subroutine available in SLAM.

A problem arose, however, with the scheduling of events from subroutine STATE. Since this subroutine is only required to be called a minimum of once each time step, a delay often arose between the time an event should have been called and the time that subroutine STATE was called and the resulting call to the event was made. Critical delay errors resulted during the calculation of aircraft position at scheduled impact time. Various time step sizes were used to reduce these errors without incurring unreasonable computer run times. A time step size of 1.0 second resulted in the omission of several events. Step sizes of .25 seconds and .1 seconds resulted in all events occurring but significant errors in the times that detonation events were called. Increments of .001 seconds resulted in satisfactory output but unsatisfactory computer run time. Finally, .05 seconds was tested and yielded a satisfactory tradeoff between the timing error in the call to the detonation events and the computer run times. As a result, time steps of .05 seconds are used in the model.

The second observation was the requirement to allow sufficient tolerance for navigation point passage by the aircraft. To reduce computational time, heading and pitch adjustments were only made at navigation points rather than continuously. It was found that even with one midcourse correction during the long ingress portion of the profile, system errors resulted in aircraft missing navigation points in excess of 100 feet laterally. While this is not a significant error in the real world, the error required the use of a 200 foot lateral tolerance for navigation point passage.

A third observation involved nonjamming aircraft and the logic required to call subroutine BREAK3. Use of the equation form of the SAM Pk assessment with a nonjamming aircraft always resulted in a very high Pk of 0.99. For a nonjamming aircraft, therefore, the driving force as to whether or not the pilot executes a break against a missile is whether or not he is aware of the missile. The nonjamming pilot, therefore, always executes a break if he is aware of the missile.

Validation

Validation of a model is a determination of whether or not the model's output is representative of the system being modeled (Ref 13:334). Law and Kelton describe the following three critical aspects of model validation (Ref 13:338-343):

1. Develop a model with high face validity.
2. Empirically test the assumptions of the model.
3. Determine how representative the simulation output data is.

To insure high face validity, extensive use was made of conventional weapons delivery literature. The logic used to model target damage was based on the highly credible Joint Munitions Effectiveness Manual. The weapons release parameters were based on parameters used by operational units (Ref 12). A USAF Tactical Fighter Weapons School instructional text was used as reference for delivery tactics (Ref 23). The threat capabilities resulted from analysis performed on three previous Air Force Institute of Technology research efforts.

The judgement of aircrew members with experience in the air-to-ground environment was used to assess the validity of the assumptions used in this research. These assumptions were considered valid and reasonable in view of the scope of this research effort.

Finally, in order to investigate whether or not the output data was representative of the system being modelled, a modified Turing test was used. In a Turing test, "experts" knowledgeable about the system being modelled are asked to differentiate between actual system data and model output data. The more difficult it is for the experts to differentiate, the greater the model's validity. A modified Turing test was used in this research as actual combat data for the scenario modelled was not available.

The experts consulted during this test were four highly experienced crewmembers with extensive air-to-ground experience in the F-4 and F-111 aircraft. All four had served as instructors and two had served as flight examiners in these aircraft. In addition, two of the four were USAF Tactical Fighter Weapons School graduates and one had been an instructor at that school. Output from the model closely approximated the expectations of these experts. They concluded, given the scenario, that the output's target damage and survivability were realistic for each of the three possible flight tactics and two weapons loads.

Summary

The verification of this model began in the earliest stages of this research. Logic was verified prior to its development into computer code; subroutines were verified prior to incorporation into the general model; traces and PRINT statements were used to verify system status as events occurred; model output results were compared to hand calculated results; and model results were analyzed graphically. Extensive verification of the general model was performed in two phases. The first phase verified the accuracy with which aircraft were moved through the target area. In the second phase, threat logic including detection and engagement of the aircraft and aircraft defensive logic were verified. The validation process incorporated a modified Turing test to evaluate model output data. Crewmembers with extensive experience in the low level air-to-ground environment validated the results of the model. The verification and validation results were satisfactory.

V. Data Analysis

This chapter involves a description of the design used to obtain data from the model. The design used both one tactic and 16 treatments and the selection of one combination of treatments applied across six different tactics. The measures of merit for each treatment are the probability of survival for the mission, which involves two aircraft attacking an airfield target, and the probability of damage that these two aircraft can achieve on the target.

Experimental Design

Subjective screening was used to select factors which should have a significant influence on mission survivability and mission target damage. Recognition of the strengths of the model and a discussion with other analysts who were familiar with the offensive counter-air mission resulted in the selection of four factors: aircraft jam capability, pilot awareness, SAM site track and acquisition time, and probability of target detection at the release point.

These four factors were analyzed using a full factorial design,'with each factor evaluated at two levels. This required a 2^4 design for the 16 treatments to be analyzed. Since the model was deterministic, only one replication per treatment was performed. The 16 combinations of these treatments are illustrated in Table V along with results obtained from the model. The analysis of these factors

TABLE V

Results of One Tactic Across 16 Combinations

JAM	YES				NO			
AWARE	YES		NO		YES		NO	
TRACK/ ACQUISITION	10,17	5,8	10,17	5,8	10,17	5,8	10,17	5,8
Prob. of Detection	.9257	.9356	.8982	.8982	.0094	.9356	5E-12	4E-12
Factor (1)	.0632	.0632	.0632	.0632	.0632	.0632	.0632	.0632
Prob. of Detection	.9257	.9356	.8982	.8982	.0094	.9356	5E-12	4E-16
Factor (.5)	.0317	.0317	.0317	.0317	.0317	.0317	.0317	.0317

TABLE VI

Results of One Cell Across Six Tactics

		TACTIC		
Bomb Type		Level Level	Level Pop	Pop Pop
High	Ps	.9367	.9350	.9239
Drag	Pd	.0260	.0632	.0971
Low	Ps	.9324	.9429	.9556
Drag	Pd	.0354	.0755	.1121

should provide sufficient insight to draw inferences about the system behavior.

The two levels selected for aircraft jamming were jam-on and jam-off. It was expected that the jam-on capability would increase mission survivability and target damage since jamming affects the kill capability of the SAM. The greater the probability of an aircraft surviving to the release point; the greater the probability of target damage when other factors are constant.

Pilot awareness was investigated at two levels, aware and not aware. It was expected that a pilot who was aware of the SAM would maneuver the aircraft to avoid missile impact, thereby increasing survivability. It was also expected that target damage might be adversely effected because of awareness. Prior to bomb release, a pilot who was aware of a missile whose Pk exceeded the maximum Pk which he could tolerate, would jettison the bombs, take evasive action against the missile, and thus never reach the target.

The high levels of SAM site track and acquisition time were 10 seconds for SAM site 1 and 17 seconds for SAM site 2. The low levels were 5 seconds and 8 seconds respectively for site 1 and site 2. Survivabilitity and damage were expected to increase as the track and aquisition time went from low to high. The longer track and acquisition time would permit the aircraft to attack the target and exit the immediate area prior to SAM launch.

The probability of seeing the target was a factor which might further reduce the JMEM based probabiliiy of target detection at the release point. This would account for situations in which the pilot had less than average capabilities. Inexperience, fatigue, or distraction due to threats could result in such situations. The high level was set at 1.0, that is, the whole value of the JMEM based probability of seeing the target. The low level of the factor was arbitrarily set at 0.5, which would reduce the probability of seeing the target by 50 percent. It was expected that this factor would effect target damage only.

One combination of treatment was then selected and applied across the six tactics in order to investigate trends with respect to survivability and damage. The cell selected was the one where jamming was on, the pilot was aware, the track and acquisition times were 5 and 8 seconds respectively for the SAM sites, and the factor for probability of target detection was 1.0. These levels were selected because they were the most likely tactical situations to exist in modern warfare. The damage was expected to increase as the delivery changed from level to angular and as the weapons changed from high drag bombs to low drag bombs. Conversely, the probability of survival was expected to decrease as the delivery changed from level to angular due to the increased delivery altitudes and time above the 100 foot minimum engagement altitude. Likewise, the use of low drag bombs was expected to reduce the survivability below that obtained with high drag bombs because of the increased delivery altitudes required for low drag munitions. The combinations and results are shown in Table VI.

Data Analysis

The data was analyzed using a confidence level of $\alpha=.10$ due to the small fixed number of observations (16). Analysis of variance was used to test the hypothesis that the specified factor had no effect. The ANOVA was performed to determine main effect trends across the 16 combinations and the differences of two-way interactions across these same combinations. Included also in the analysis was the determination of differences across the six tactics.

The trends of the two responses of survivability and damage across the 16 combinations are illustrated in Figure 17. Because of the specified scenario, the SAMs always intercepted the aircraft after weapons release. Hence, the only factor affecting probability of target damage was the target detection factor. For this same reason, the factors of jamming, awareness, and track and acquisition times had no effect on the probability of arrival or the resulting target damage. Depicted in Figure 17, therefore, are the four treatment effects which produced a statistically significant change in the response variable. The specific response variable values are in Appendix E.

In the case of the effect of detection on the target damage response variable (Fig. 17(d)), a decrease in damage resulted from a decrease in the factor level. This trend was

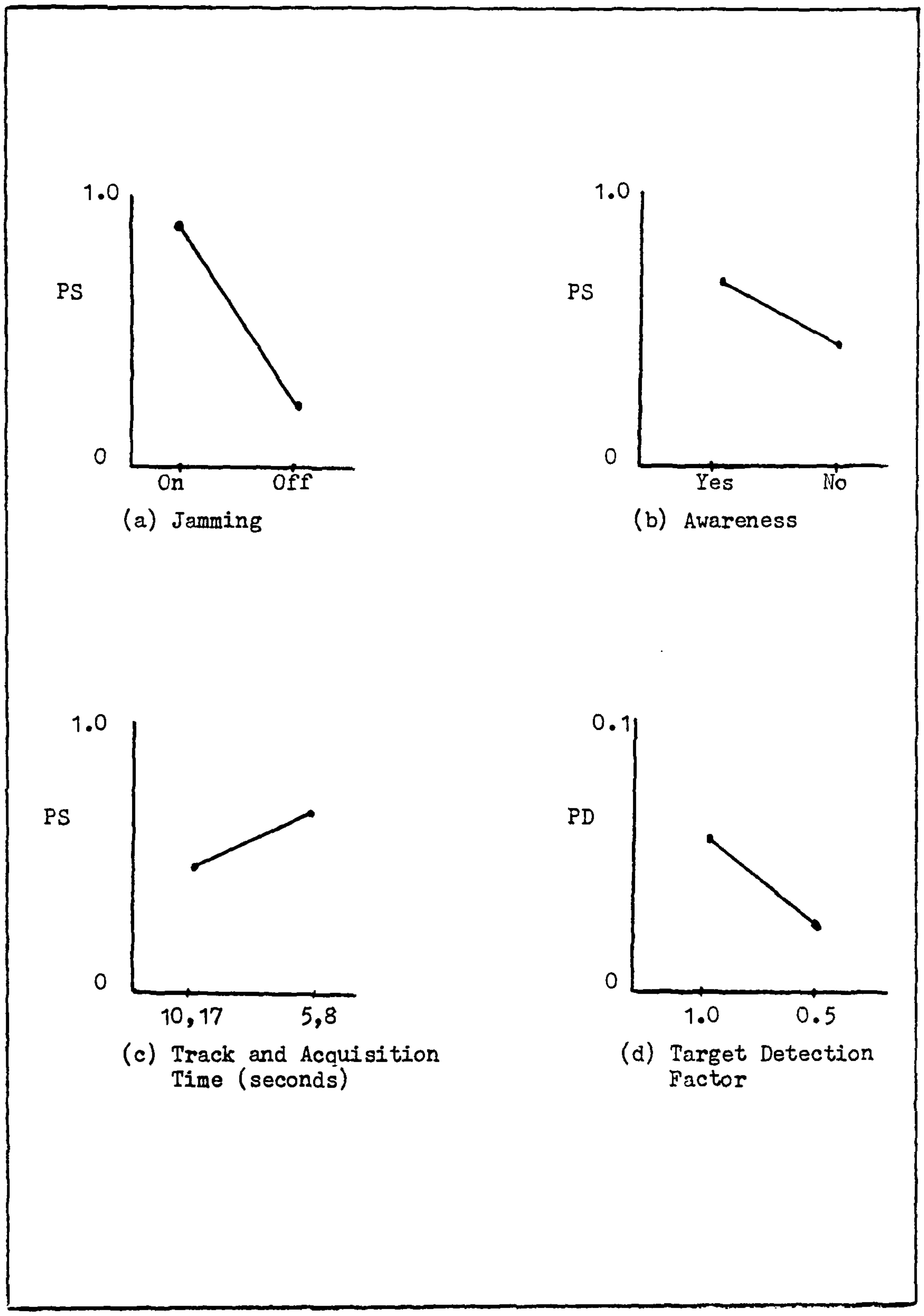


Fig. 17. Main Effects

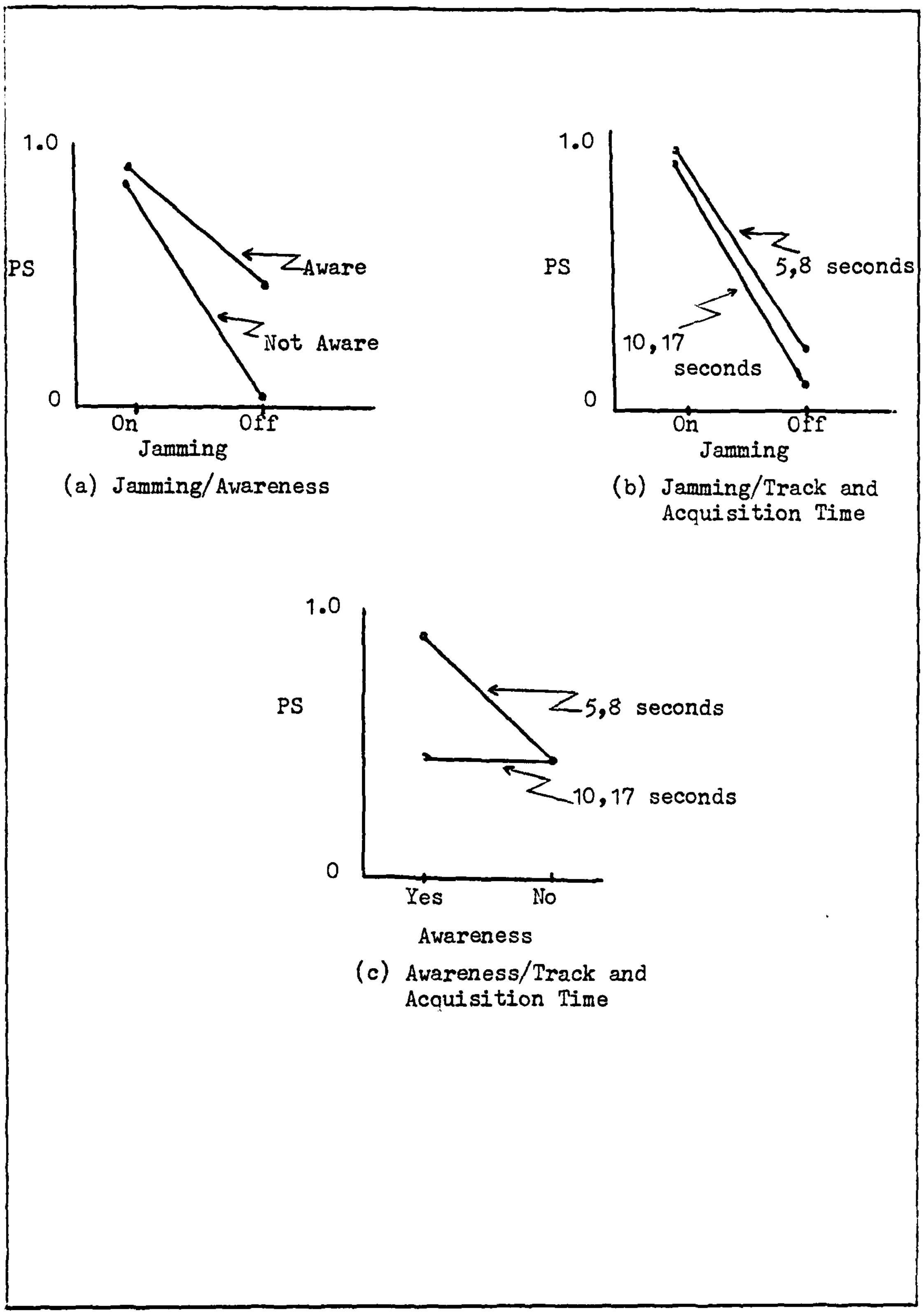
found to be significant with a probabiliity of detection value equal to 0.5. The null hypothesis of no treatment effect was therefore rejected for an α level of .10.

The three main effects on survivability depicted in Figure 17 also indicated significant trends. Figure 17(a) shows that a jamming aircraft has a survival advantage over the nonjamming aircraft. This was as expected. The trend was significant with a p-value of .0019. Pilot awareness (Fig. 17(b)) also resulted in a significant increase in survival as this awareness permitted evasive action by the pilot. The probability did change significantly with changed awareness as indicated by a p-value of .0561. It was also recognized that the null hypothesis would not have been rejected had the confidence level been set at .95. Finally, the affect of track and acquisition time depicted in Figure 17(c) indicates an increase in survivability as track and acquisition time is decreased from 10 and 17 seconds to 5 and 8 second. The trend was significant (p-value=.07) but not expected.

A thorough investigation of the model's results indicated that the track and acquisition times affected the final intercept geometry which, in turn, affected the probability of aircraft survival. Given track and acquisition times of 5 and 8 seconds for the sites, aircraft number one performed two defensive maneuvers. The first maneuver was performed during the aircraft turn off target. The aircraft's heading following this successfull maneuver resulted in an approximate ninety degree intercept angle with missile number two. Once again, aircraft number one successfully maneuvered against the missile threat. The geometry was different, however, for the track and acquisition times of 10 and 17 seconds. With these longer times, the first maneuver of aircraft number one was successful, but the post maneuver turn to exit the target area resulted in a frontal intercept with the second missile. During the defensive maneuver against the second missile, the aircraft was unable to sufficiently move out of the plane of the attack. This resulted in a low probability of survival for the aircraft. Hence, this result appears to be scenario dependent.

The analysis also investigated interactions of the main effects. Figure 18 illustrates the combined effects of the three significant factors on the survivability response variable . (Specific values are found in Appendix F.) The scenario results indicated that the combination of target detection with any of the other three factors did not produce a significant interactive effect in probability of aircraft survival. The other three factors did, however, interact with each other to produce differences in survivability.

Figure 18(a) shows that if the aircraft was not jamming and the pilot was not aware of the missile threat his chances of survival were minimal. The probability of survival drastically increased if the aircraft was jamming even though the pilot was still not aware of the missile threat. Pilot



(a) Jamming/Awareness

(b) Jamming/Track and Acquisition Time

(c) Awareness/Track and Acquisition Time

Fig. 18. Interactions

awareness increased survivability for the non-jamming aircraft. However, awareness increased survivability to a much lesser extent in the jamming situation.

These results do not indicate that awareness should be sacrificed for jamming but rather, given an effective jamming system the requirement for total awareness can be reduced without a great change in mission survivability. The illustrated difference of combined jamming and awareness was statistically significant (p=.0834).

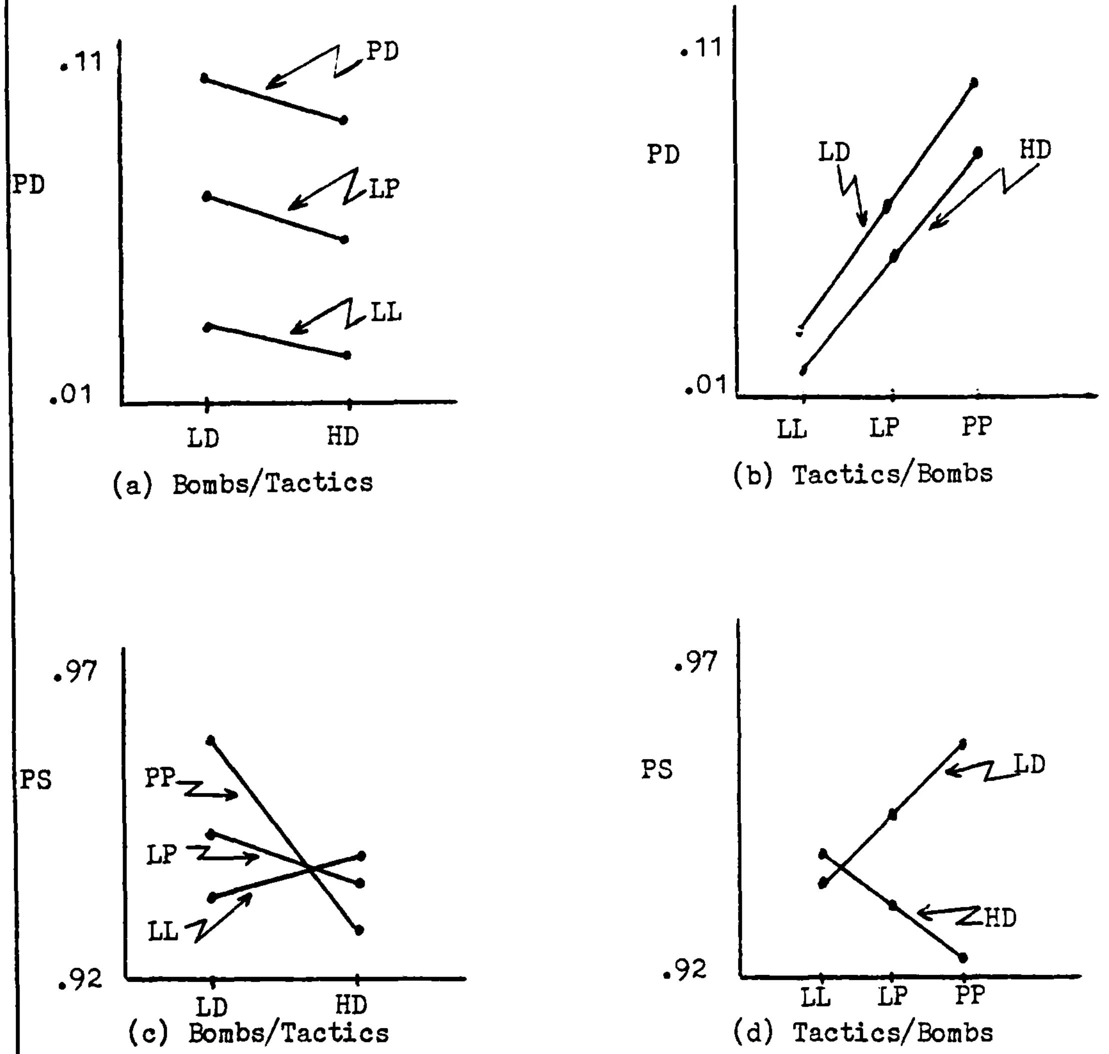
Figure 18(b) shows the interaction of jamming with track and aquisition time. In the non-jam situation, the shorter track and acquisition time result in higher survivability. This counter-intuitive result is attributed to the intercept geometry of the shorter track and acquisition times. The difference in survivabiliity is negligible for the two levels of track and acquisition time with a jamming aircraft. The interaction effect between jamming and the track and acquisition times is statistically significant (p=.0745).

Figure 18(c) demonstrates the differences in survivability caused by the combined effect of awareness and track and acquisition times. When the pilot was not aware of the missile, the probability of survival was the same for both levels of track and acquisition times. However, when the pilot was aware of the missile threat the shorter track and acquisition times produced a greater probability of survival. This difference was statistically significant (p=.0701). As described earlier, this was due to the strong effect of intercept geometry on survivability and the problems associated with taking evasive action on two successive missiles.

The 16 combinations of treatments used conjunction with the level/pop tactic and high drag bombs identified three treatments which affected survivability and one treatment which affected damage. A similar ANOVA was performed on the results of the one combination of the treatments over the six previously identified tactics.

Figure 19 illustrates the trends across the six different tactics as bombs change from low drag to high drag and as the delivery mode goes from level to angular. The results for the probability of damage were as expected. Figure 19(a) illustrates that as bombs were changed from low drag to high drag there was a drop in damage for the same tactic. The main effect was statistically significant (P=.00137), however, there was no interaction. Figure 19(b) shows the increase in damage across tactics when the weapon is kept the same. This difference is accounted for by the different circular error probable (CEP) associated with the delivery mode. This was significant (P=.0013), again, there was no interaction.

Figures 19(c) and 19(d) both show results that were not expected. When compared to high drag bombs, low drag bombs required a higher delivery release altitude for the same tactic. This exposed the aircraft to the threats for longer periods of time. As a result, this higher exposure time was



where

LL	= Tactic of lead aircraft level; wingman aircraft level
LP	= Tactic of lead aircraft level; wingman aircraft angular
PP	= Tactic of lead aircraft angular; wingman aircraft angular
LD	= Low drag bomb
HD	= High drag bomb
PS	= Probability of aircraft survival
PD	= Probability of target damage

Fig. 19. Trends Across Tactics

expected to decrease survivability. Similarly, the pop-to-angular deliveries were expected to have lower survivability than level deliveries due to the increased exposure time associated with the angular deliveries. It was expected, therefore, to produce a decrease in survivability, as tactics changed from level to angular.

Figure 19(c) shows a greater survivability for low drag bombs than for high drag bombs when using the level/pop and pop/pop tactics. The unexpected increase in survivability for low drags was due to the dynamics of the system modeled. The results demonstrate that the model does not restrict itself solely to the interaction between one aircraft and one threat at any point in time. Rather, the results indicate the complexities of multiple threats engaging an aircraft, an aircraft which may or may not be actively maneuvering against one of the threats.

The increased survivability was often related to aircraft position and g-loading when engaged by AAA sites. In the level/pop low-drag delivery, aircraft one had just finished maneuvering against the first missile when site AAA 3 began firing and was, therefore, at a greater slant range from the AAA site than for the high drag bomb case. In the low drag case, an earlier climb to release altitude was required. This permitted earlier acquisition and tracking by the SAM site and, thus earlier SAM engagement. Aircraft two also contributed to the higher survivabiliity in the low drag case. The earlier pull-up point caused a 4g condition at the time AAA site 4 began firing, resulting in a zero Pk assessment. AAA site 3 also had a zero PK assessment on aircraft two because of aircraft two's 8G break against missile 3 at AAA 3 fire time. The longer time over target caused aircraft two's arrival at the AAA 3 fire point to coincide with missile intercept instead of just doing a 4g turn off target as in the high drag case.

The increased survivability noted in the low drag pop/pop delivery (Fig 19(c)) was attributed to both aircraft one and two. Aircraft two's contribution was the same as in the level/pop tactic previously described. Aircraft one's earlier pull-up in the low drag case caused AAA 4 to fire during the climb to the rollin point. In the high drag case AAA 4 fired on aircraft one just prior to the aircraft commencing the pull-up. In the low drag case, therefore, the aircraft was at a higher altitude and longer slant range from the AAA site than the aircraft in the high drag case. These differences in survivability as the bomb type was changed with respect to tactic were statistically significant (P=.0137).

Figure 19(d) shows that survivability increased as tactics went from level to angular using the low drag bomb. The increase in survivability was due to aircraft position. Both aircraft one and two passed closer to AAA 4 during pop-to-angular deliveries than for the level deliveries. The site fired on each aircraft after the start of the pull-up for bomb delivery. This produced a lower survivability as the aircraft ground track approached the AAA site. The survivability, therefore, increased because of a 4G climb and not solely as a function of time above 100 feet altitude. These occurrences combined to produce the unexpected trend in survivability. The differences in survivability were due to interactions of the variables and not significant main effects.

Summary

This chapter has shown that the factors of aircraft jamming, pilot awareness and SAM site track and acquisition time affect the mission survivability. Statistically significant trends were shown for these main effects and for interactions between these main effects. In addition, because the scenario resulted in missile intercept of aircraft only after bomb release, the probability of target detection had the only significant effect on the target probability of damage for the mission. This effect was statistically significant. With respect to the six tactics, the probability of damage trends occurred as expected. There were, however, unexpected results in survivability. These were explained through investigation of the actual encounter conditions. Although the probability of survival changes were small for the different mixes of bombs and tactics (.92 to .95) the results indicated an interaction between tactics and bomb types.

It is recognized that the results investigated are extremely scenario dependent. Further investigation should be made to determine the effects that a change in scenario might have on the measure of merit. In addition, a full factorial design, including all treatments identified in the offensive counter mission, should be conducted to determine all main effects and interactions in the mission.

VI. Summary, Conclusions and Recommendations

Summary

The primary objective of this thesis, as stated in Chapter I, was to develop a methodology which could be used to analyze the likely target damage and resulting survival of friendly aircraft in the offensive counterair mission. It was decided that the methodology should interface with the weapons effects calculations of the highly credible and widely used Joint Munitions Effects Manual (JMEM). In addition, two topics not addressed in JMEM calculations needed to be included in the methodology: the probability of aircraft arrival at the target and the probabiliy of aircraft survival.

The scenario chosen for this research was a mission of two aircraft attacking a specific area target at an enemy airfield. The area of operations, the target area, was a circle of ten nautical mile radius centered on the airfield's runway. Located within this area were specifically defined AAA and SAM threats.

Four basic steps were chosen to accomplish the objectives of the research. First, the offensive counterair mission had to be planned with consideration given to the type of weapons and tactics used as well as the enemy threats. Second, the probability of target damage from each aircraft needed to be calculated. Third, the survivability of each aircraft needed to be assessed. Finally, the individual probabilities of target damage and aircraft survival needed to be combined into overall mission probabilities of target damage and aircraft survival.

It was decided that these requirements could best be satisfied by incorporating an analytical weapon's effects routine into a discrete-continuous simulation model. The analytical routine was based on a hand calculation methodology available in JMEM. Simulation was primarily used to model the routing of aircraft through a target area of ground based threats and to study the resulting survival of those aircraft.

SLAM was chosen as the simulation language because of the capabilities of SLAM to model discrete-continuous systems. While the continuous capabilities of SLAM were used extensively, the dependence on the SLAM discrete logic and file structures was minimized so as to reduce the understanding of the SLAM language needed to use the methodology.

The verification and validation of this model began in the earliest stages of its development. The model was created in modules. High face validity was stressed during the creation of each module. Each of these modules, or subroutines, was then verified individually prior to its incorporation into the main model. Once the development of the main model was complete, the accuracy with which each aircraft moved through the target area and the execution of the threat logic were extensively verified. Finally, the output from the fully developed main model was verified by a panel of tactical aircrew members using a modified Turing test.

Once the model was completely developed, verified, and validated, analysis was conducted. However, the actual level of analysis performed was extremely limited due to time constraints. The analytical methods and limited results described in Chapter V were credible, but general conclusions or recommendations should not be drawn from these highly scenario dependent results due to the limited sample size and lack of extensive sensitivity analysis.

This thesis effort, therefore, resulted in the development of a model which can be used to evaluate the probability of target damage and resulting aircraft attrition on an offensive counterair mission. Given a user-defined scenario, the weapons effects are assessed by applying the analytical JMEM methodology, and the three-dimensional threat environment is portrayed using a SLAM discrete-continuous model.

The selection of the navigation points to the target for each aircraft is available via a flight planning routine. This routing can incorporate either a level or a pop-to-angular delivery. Once a candidate run-in heading is identified, an automated threat search is performed to select the axis of attack which minimizes exposure to threat systems. The planning routine also allows a coordinated attack of two aircraft. That is, the output of this routine reflects the deconfliction of aircraft required by weapons effects, the tactical considerations of roll-in direction, and the desired relative positions of each aircraft.

Enroute to the target, the model uses three-dimensional navigation and three-dimensisonal threat encounter geometry. These calculations permit positioning of each aircraft and the determination of aircraft vulnerabilities to various threats. The pilot's awareness of threats is also incorporated. Given a detected missile, the model allows a decision by the pilot whether or not to maneuver against the missile.

During the missile's intercept, collision course steering is effectively modelled via predicted impact point updates. The inabiliity of a missile to correct its planned impact point in response to a well executed aircraft defensive maneuver is also modelled. The assessment of probability of kill by a SAM against either a maneuvering or nonmaneuvering aircraft is possible via a 109 cell probability of kill logic. The assessment of probability of kill due to engagement by an AAA site is available in a short analytical routine.

Conclusion

This thesis resulted in the development of an effective methodology which can be used to assess target damage and friendly aircraft survivability in the offensive counterair mission. The methodology's output of damage and survivability can then be graphically displayed for decision-making trade-off studies.

Recommended Follow-On Study

The following are possible follow-on areas of study:

1. Various attack directions should be investigated to determine if survivability can be appreciably enhanced without a significant change in target damage. Different attack axes would change the effective target dimensions.

2. Various target locations should be investigated using the same threat placement. This would determine an area about the runway which could be attacked without a significant effect on aircraft survivability given defined threat locations.

3. Various threat locations should be investigated to determine an optimum placement which would enable aircraft intercept prior to bomb release. This may present new ideas in placement of our own airfield defensive systems.

4. Enhance threat capabilities to include multiple AAA firings, handheld infrared missile locations based on a probabilistic distribution, and changes in SAM velocity and electronic counter-countermeasure capabilities. This would further identify survivability sensitivities to threat capabilities.

5. Incorporate lethal missile volume calculations which determine the probability of kill based on engagement conditions. This would include a calculation of the fuzing

errors and warhead flyoff angles and velocities to more accurately identify the kill capabilities of a SAM in the dynamic engagement situation.

6. Further enhance the aircraft evasive maneuver by the addition of a positioning maneuver prior to the break turn, and, if altitude permits, an aircraft descent during the break turn. An airspeed decrease could also be added during the break to model those aircraft which historically lose airspeed during high g turns (F-4, F-111, etc.).

7. Develop an experimental design incorporating all the factors identified in Figures 6 and 7 in Chapter II. This would include the addition of stochastic parameters on applicable factors (track and acquisition time, pilot awareness, etc.).

Bibliography

1. Adams, Marvis R. AFATL/DLYW. Telephone conversation. Eglin Air Force Base, Florida 16 February 1984.

2. AFM 1-1. Functions and Basic Doctrine of the United States Air Force. Washington, D.C.: Department of the Air Force, February 1979.

3. Anderson, Kenneth C. and Ronald B. Nenner. "A Wild Weasel Penetration Model." Master of Science Thesis. School of Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, March 1982.

4. Breuer, Walter. Lecture material for SE 5.81, Survivability and Vulnerability of Systems (Conventional). School of Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 1983.

5. Bridgeman, Charles J. Lecture material for NE 6.95, Nuclear Systems Survivability. School of Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, 1983.

6. Callero, Monti et al. Toward an Expert Aid for Tactical Air Reporting. Report on use of knowledge engineering in tactical planning problems. Arlington, Virginia: Defense Advanced Research Projects Agency, Department of Defense, January 1981.

7. Emerson, D.E. AIDA: An Airbase Damage Assessment Model. Report for the Director of Planning, Programming, and Analysis, HQ USAF. Santa Monica, California: The Rand Corporation, September 1976.

8. Golden, Major August. Radar Electronic Warfare. Wright-Patterson Air Force Base, Ohio: Department of the Air Force, Air Force Institute of Technology, May 1983.

9. Goodson, Brig. Gen. Wilfred L. Former Deputy Chief of Staff for Plans, United States Air Forces Europe (USAFE). Briefing to graduate class of Strategic and Tactical Science Program, Air Force Institute of Technology Wright-Patterson Air Force Base, Ohio, 21 April 1983.

10. Hillier, Frederick S. and Gerald J. Lieberman. Introduction to Operations Research (Third Edition). San Francisco, California: Holden-Day Inc., 1980.

11. LAKP 51-7. Flying Training Weapons Delivery. Training pamphlet. RAF Lakenheath, England: United States Air Force Europe (USAFE), October 1979.

12. LAKP 60-2. Flying Tactical Aircrew Aid. RAF Lakenheath, England: United States Air Forces Europe (USAFE), February 1981.

13. Law, Averill M. and W. David Kelton. Simulation Modeling and Analysis. New York: McGraw-Hill Book Company, 1982.

14. Leek, Warren J. and Richard W. Schmitt. "Survivability Study of a FLIR Equipped Fighter on a Night Penetration of a Soviet Army." Master of Science Thesis, School of Engineering, Air Force Institute of Technology, Wright-Patterson Air Foce Base, Ohio, March 1981.

15. Miller, John M. "Low Angle Low Drag". USAF Fighter Weapons Review, 26-34 (Summer 1981).

16. Neal, Donald W. and Gary G. Kizer. "A Simulation Model to Evaluate Close Air Support Kill-to-Loss Ratios". Master of Science Thesis. School of Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio, March 1983.

17. Pritsker, A. Alan B. and Claude Dennis Pegden. Introduction to Simulation and SLAM. New York: John Wiley and Sons, 1979.

18. Quattromoni, LtCol Anthony F. Catalog of Wargaming and Military Simulation Models. Washington D.C.: Studies, Analysis, and Gaming Agency; Organization of the Joint Chiefs of Staff, May 1982.

19. Shannon, Robert E. Systems Simulation: The Art and the Science. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1975.

20. "Classified Document: Qualified requestors may obtain this reference from AFIT/ENS, Wright-Patterson AFB OH 45433."

21. "Classified Document: Qualified requestor may obtain this reference from AFIT/ENS, Wright-Patterson AFB OH 45433."

22. Thierauf, Robert J. and Richard A. Grosse. Decision Making Through Operatins Research. New York: John Wiley and Sons, Inc., 1972.

23. USAF Fighter Weapons School. Instructional Text Weapons Delivery F-111, Vol I. Text for course F-111 OIDOAI/DOWI, Nellis Air Force Base, Nevada: Tactical Air Command, September 1981.

APPENDIX A

NON-SLAM VARIABLE LISTING

This appendix contains definitions of non-SLAM variables. SLAM variables are defined within the computer code listed in Appendix B.

-A-

A	=	(PKSAM9)	Missile constant
A6	=	(AREA5)	Limit of integration
AA2	=	(MBRAN2)	Aspect of point B from point A
AA3	=	(BREAK3)	Aspect of missile to aircraft
AA4	=	(SCAN4)	Aspect of threat site to center of target
AA7	=	(ROUTE7)	Aspect of runway from aircraft at climb point
AA9	=	(PKSAM9)	Aspect of aircraft to planned impact point
AAREA	=	(AREA5)	Result of numerical integration
AB	=	(JMEM6)	Single weapon effective area (square feet)
ACFTX8	=	(IMPCT8)	Aircraft current X coordinate
ACFTY8	=	(IMPCT8)	Aircraft current Y coordinate
ACFTZ8	=	(IMPCT8)	Aircraft current Z coordinate
ADIV6	=	(JMEM6)	Absolute value of dive angle
ADIV7	=	(ROUTE7)	Absolute value of dive angle
AET	=	(JMEM6)	Effective target area (square feet)
ALFA1	=	(XYCOR1)	Angle between X axis and line from A to B
ALFA2	=	(MBRAN2)	Angle between X axis and line from A to B
ALFA7	=	(ROUTE7)	Angle at climb point between run-in and runway
ALTIN7	=	(ROUTE7)	Ingress altitude prior to climb (feet)
ANG	=	(ROUTE7)	Angle used in calculating roll-in (degrees)
AP	=	(JMEM6)	Effective stick pattern area (square feet)
APXALT	=	(ROUTE7)	Apex altitude during angular delivery (feet)
AV10	=	(PKAA10)	Average aircraft vulnerable area (square meters)
AWARE	=	(BREAK3)	Awareness of pilot of missile: 1=aware; 0=not aware
AX1	=	(XYCOR1)	Known X coordinate of point A
AX2	=	(MBRAN2)	Known X coordinate of point A
AY1	=	(XYCOR1)	Known Y coordinate of point A
AY2	=	(MBRAN2)	Known Y coordinate of point A
AZ2	=	(MBRAN2)	Known Z coordinate of point A

-B-

B	=	(ROUTE7)	Radius from runway center to aircraft flight path (feet)
B	=	(PKSAM9)	Missile constant
B6	=	(AREA5)	Limit of integration
BAREA	=	(AREA5)	Result of numerical integration
BOOM9	=	(PKSAM9)	Flag: 1=detonation; 0=predetonation
BRAKPK	=	(BREAK3)	Missile pk at which pilot will break
BX1	=	(XYCOR1)	X coordinate of point B
BX2	=	(MBRAN2)	X coordinate of point B
BY1	=	(XYCOR1)	Y coordinate of point B
BY2	=	(MBRAN2)	Y coordinate of point B
BZ2	=	(MBRAN2)	Z coordinate of point B

-C-

C	=	(ROUTE7)	Radius of target area (feet)
C	=	(PKSAM9)	Missile constant
C6	=	(AREA5)	Input constant
CEP	=	(JMEM6)	Bomb circular error probable ground plane (feet)
CEPF	=	(PKSAM9)	Missile circular error probable (feet)
CEPM	=	(PKSAM9)	Missile circular error probable (meters)
CLMANG	=	(ROUTE7)	Climb angle as aircraft departs ingress altitude
CNTLN	=	(SCAN4)	Orientation of length of target area (degrees)

-D-

D	=	(AREA5)	Temporary variable
D	=	(ROUTE7)	Ground distance along flight path from climb point to closest point to runway center
D	=	(PKSAM9)	Missile constant
D7	=	(ROUTE7)	Angle between ingress heading and run-in heading
DACFTX	=	(IMPCT8)	Change in aircraft X coordinate during time-to-impact
DACFTY	=	(IMPCT8)	Change in aircraft Y coordinate during time-to-impact
DACFTZ	=	(IMPCT8)	Change in aircraft Z coordinate during time-to-impact
DEGTT	=	(ROUTE7)	Degrees to turn during roll-in
DEP	=	(JMEM6)	Deflection error probable (feet)
DIV6	=	(JMEM6)	Aircraft dive angle (degrees)
DIV7	=	(ROUTE7)	Aircraft dive angle (degrees)

DMSLX	=	(IMPCT8)	Missile X coordinate change during time-to-impact
DMSLY	=	(IMPCT8)	Missile Y coordinate change during time-to-impact
DMSLZ	=	(IMPCT8)	Missile Z coordinate change during time-to-impact
DRAW1	=	(EVENT)	Random variable to assess pilot #1 missile awareness
DRAW2	=	(EVENT)	Random variable to assess pilot #2 missile awareness
DX1	=	(XYCOR1)	Change in X coordinates between points A and B
DX2	=	(MBRAN2)	Change in X coordinates between points A and B
DY1	=	(XYCOR1)	Change in Y coordinates between points A and B
DY2	=	(MBRAN2)	Change in Y coordinates between points A and B
DZ2	=	(MBRAN2)	Change in Z coordinates between points A and B

-E-

E	=	(ROUTE7)	Ground distance from entry point along flight path to point closest to runway center
E	=	(PKSAM9)	Missile constant
EFD	=	(JMEM6)	Effective target coverage in width (percent)
EFR	=	(JMEM6)	Effective target coverage in range (percent)
EI	=	(JMEM6)	Effectiveness index
ERPDB	=	(PKSAM9)	Effective radiated power (db)
ERROR	=	(AREA5)	Numerical integration tolerance
ESUM	=	(AREA5)	Sum of numerical integration

-F-

F	=	(ROUTE7)	Ground distance from entry point to climb point (feet)
F	=	(PKSAM9)	Missile constant
FAC	=	(PKSAM9)	Probability that missile lands in a cell
FCTRL	=	(AREA5)	Temporary variable
FCTRU	=	(AREA5)	Temporary variable
FCTX	=	(AREA5)	Area of one interval in numerical integration
FLAG8	=	(IMPCT8)	Flag: 1=aircraft outrunning missile
FNLLD4	=	(SCAN4)	Run-in heading for aircraft number one
FNLWG4	=	(SCAN4)	Run-in heading for aircraft number two

-G-

G1	= (---)	G load aircraft number one
G2	= (---)	G load aircraft number two
GR	= (---)	Ground range in feet
GR1	= (XYCOR1)	Ground range point A to point B (feet)
GR4	= (SCAN4)	Ground range target center to threat site (feet)
G.?6	= (JMEM6)	Ground range of first bomb released (feet)
GRDB	= (PKSAM9)	Threat radar gain (db)
GRESS6	= (JMEM6)	Temporary variable
GRIMP	= (EVENT)	Ground range of threat site to impact point (feet)
GRIMP8	=)IMPCT8)	Ground range of threat site to impact point (feet)
GRI	= (JMEM6)	Ground range from first weapon release to target (feet)
GZX9	= (PKSAM9)	X coordinate (impact point system) of cell center
GZY9	= (PKSAM9)	Y coordinate (impact point system) of cell center
GZZ9	= (PKSAM9)	Z coordinate (impact point system) of cell center

-H-

HDG2	= (MBRAN2)	Heading of entity of point A
HDG8	= (IMPCT8)	Required missile heading for intercept
HDGIN7	= (ROUTE7)	Temporary variable
HEAD1	= (---)	Heading flag aircraft one: 1=change required
HEAD2	= (---)	Heading flag aircraft two: 1=change required
HT	= (EVENT)	Height (feet)
HYPOT	= (ROUTE7)	Range from roll-in point to track point (feet)

-I-

I5	= (AREA5)	Counter for five intervals in numerical integration
IA	= (JMEM6)	Impact angle (degrees)
IMPTX8	= (IMPCT8)	Predicted impact point's X coordinate
IMPTY8	= (IMPCT8)	Predicted impact point's Y coordinate
IMPTZ8	= (IMPCT8)	Predicted impact point's Z coordinate
IMSL	= (EVENT)	Missile number (1,2,3,or 4)
IMSL3	= (BREAK3)	Missile number
INTV	= (JMEM6)	Intervolometer setting (seconds)

ITHRT	= (EVENT)	Missile number

-J-

JAM9	= (PKSAM9)	Jamming option: 1=jamming; 0=no jamming
JS	= (PKSAM9)	Jamming to signal ratio of threat radar
JSDB	= (PKSAM9)	Jamming to signal ratio (db)

-K-

K	= (EVENT)	Counter to indicate which aircraft engaged
KNTR1	= (EVENT)	Counter indicates navigation points for aircraft one
KNTR2	= (EVENT)	Counter indicates navigation points for aircraft two
K5	= (AREA5)	Temporary variable

-L-

LA	= (JMEM6)	Target length (feet)
LB	= (JMEM6)	Single weapon effective length (feet)
LEP	= (JMEM6)	Effective pattern length (feet)
LET	= (JMEM6)	Effective target element length (feet)
LP	= (JMEM6)	Effective stick pattern length (feet)
LS	= (JMEM6)	Stick length (feet)
LSK	= (JMEM6)	Stick length factor
LR	= (PKSAM9)	Lethal radius (feet)

-M-

MAG	= (PKSAM9)	Slant range from aircraft to detonation point (feet)
MB1	= (XYCOR1)	Magnetic bearing of point B from point A
MB2	= (MBRAN2)	Magnetic bearing of point B from point A
MB7	= (ROUTE7)	Magnetic bearing of runway center from climb point
MSLX8	= (IMPCT8)	Current X coordinate of missile
MSLY8	= (IMPCT8)	Current Y coordinate of missile
MSLZ8	= (IMPCT8)	Current Z coordinate of missile

-N-

N	= (EVENT)	Missile number
N6	= (JMEM6)	Effective number of bombs in single bomb effective area
NALFA	= (ROUTE7)	Flag used to indicate geometry
NCON6	= (SCAN4)	Number of threats in 120 degree cone prior to target
NCON12	= (SCAN4)	Number of threats in 120 degree cone beyond target
NEIT	= (JMEM6)	Effectiveness index type
NP	= (JMEM6)	Number of bombs released per release pulse
NQUAD	= (XYCOR1)	Cartesian quadrant
NR	= (JMEM6)	Number of release pulses
NSTAR	= (JMEM6)	Number of weapons in pattern
NTHRT	= (SCAN4)	Number of threat sites in target area

-P-

P6	= (AREA5)	Imput ratio
PCD	= (JMEM6)	Probability of damage within pattern
PCD6	= (JMEM6)	Conditional probability of damage
PCH8	= (IMPCT8)	Required pitch for missile to complete intercept
PDMSN	= (EVENT)	Overall mission target damage (due to both aircraft)
PHD	= (JMEM6)	Probability of damage given a hit
PHI	= (PKSAM9)	Angle formed by aircraft position at detonation time, the detonation point, and aircraft prebreak vector
PITCH1	= (---)	Pitch flag for aircraft one: 1=pitch change required
PITCH2	= (---)	Pitch flag for aircraft two: 1=pitch change required
PK9	= (PKSAM9)	Probability of aircraft kill by a SAM
PK10	= (PKAA10)	Probability of aircraft kill by AAA
POPRAD	= (ROUTE7)	Radius of turn during roll-in (feet)
PKSS10	= (PKAA10)	AAA single shot probability of kill
POPRI	= (SCAN4)	Direction of roll-in; 1=right; 2=left
POPRI4	= (SCAN4)	Same as POPRI
PRDB	= (PKSAM9)	Threat radar output (db)
PSEE	= (JMEM6)	JMEM probability that pilot sees target
PSMSN	= (EVENT)	Overall mission survivability (for both aircraft)
PZXRF	= (IMPCT8)	Change in X coordinates between aircraft and missile
PZYRF	= (IMPCT8)	Change in Y coordinates between aircraft and missile
PZZRF	= (IMPCT8)	Change in Z coordinates between aircraft and missile

-R-

R6	=	(JMEM6)	Weapon reliabilities
R9	=	(PKSAM9)	Slant range from missile site to predicted impact point
RCS	=	(PKSAM9)	Aircraft radar cross section (square meters)
RCSDB	=	(PKSAM9)	Aircraft radar cross section (db)
RD	=	(JMEM6)	Delivery reliability
RDB	=	(PKSAM9)	Decibel conversion factor
RDS	=	(PKAA10)	Number of bullets fired per AAA burst
REP	=	(JMEM6)	Range error probable
RIDIR7	=	(ROUTE7)	Roll-in direction
RL	=	(AREA5)	Lower limit of integration
RM	=	(PKSAM9)	Slant range from missile site to impact point (meters)
RNINLD	=	(SCAN4)	Run-in heading for aircraft number one
RNINWG	=	(SCAN4)	Run-in heading for aircraft number two
RU	=	(AREA5)	Upper limit of integration
RUNIN	=	(SCAN4)	Threat minimizing target attack axis (degrees)
RUNIN7	=	(ROUTE7)	Run-in heading

-S-

S1	=	(IMPCT8)	Magnitude of vector SR0 (feet)
S2	=	(IMPCT8)	Magnitude of vector SR1 (feet)
SAMTP9	=	(PKSAM9)	Type of SAM
SARC	=	(ROUTE7)	Distance flown along arc during roll-in (feet)
SB	=	(JMEM6)	Weapon spacing on ground (feet)
SIGB	=	(JMEM6)	Bomb ballistic error (milliradians)
SIGBD	=	(JMEM6)	Bomb ballistic deflection error (feet)
SIGBR	=	(JMEM6)	Bomb ballistic range error (feet)
SR01	=	(IMPCT8)	X component of SR0
SR02	=	(IMPCT8)	Y component of SR0
SR03	=	(IMPCT8)	Z component of SR0
SR11	=	(IMPCT8)	X component of SR1
SR12	=	(IMPCT8)	Y component of SR1
SR13	=	(IMPCT8)	Z component of SR1
SR2	=	(MBRAN2)	Slant range from point A to point B (feet)
SR6	=	(JMEM6)	Slant range from first bomb release point to impact point
SR9	=	(PKSAM9)	Slant range from aircraft to planned impact point (feet)
SR10	=	(PKAA10)	Slant range from AAA site to aircraft (feet)
SRT	=	(JMEM6)	Slant range from first release point to target (feet)

SRX9	= (PKSAM9)	Change in X (impact point system) between cell center and aircraft position
SRY9	= (PKSAM9)	Change in Y (impact point system) between cell center and aircraft position
SRZ9	= (PKSAM9)	Change in Z (impact point system) between cell center and aircraft position
SSPD	= (JMEM6)	Single sortie expected fractional coverage (percent)
STURN	= (MBRAN2)	Direction of shortest turn: 1=right; 2=left

-T-

T6	= (AREA5)	Input ratio
TANG	= (ROUTE7)	Temporary variable
TAREA5	= (AREA5)	Total area under curve
TAWAREE	= (AREA5)	Temporary variable
TEMP1	= (---)	Temporary variable
TEMP2	= (---)	Temporary variable
TEMP8	= (IMPCT8)	Temporary variable
TEMP90	= (---)	Temporary variable
TGTG10	= (PKAA10)	Current G load of aircraft engaged by AAA
TGTX	= (SCAN4)	Target center X coordinate
TGTY	= (SCAN4)	Target center Y coordinate
TGTZ	= (SCAN4)	Target center Z coordinate
THDG	= (ROUTE7)	Temporary variable
TOF	= (PKAA10)	Bullet time of flight (seconds)
TOFMX8	= (IMPCT8)	Maximum missile time of flight remaining (seconds)
TOTGR	= (ROUTE7)	Total ground range to target
TPCAC	= (PKSAM9)	Distance of cell center from impact point (feet)
TRKTM	= (ROUTE7)	Track time (seconds)
TTI8	= (IMPCT8)	Time-to-impact (seconds)

-V-

V6	= (JMEM6)	Aircraft velocity (knots)
V7	= (ROUTE7)	Aircraft velocity (knots)
V8	= (IMPCT8)	Missile velocity (knots)
VELX8	= (IMPCT8)	Aircraft current X component of velocity (ft/sec)
VELY8	= (IMPCT8)	Aircraft current Y component of velocity (ft/sec)
VELZ8	= (IMPCT8)	Aircraft current Z component of velocity (ft/sec)
VI	= (PKAA10)	Initial bullet speed (M/S)
VF	= (PKAA10)	Final bullet speed (M/S)

-W-

WA	= (JMEM6)	Target width (feet)
WB	= (JMEM6)	Single bomb effective width (feet)
WEP	= (JMEM6)	Effective pattern width (feet)
WET	= (JMEM6)	Effective target element width (feet)
WP	= (JMEM6)	Effective stick pattern width (feet)
WS	= (JMEM6)	Stick width (feet)

-X-

XPOS9	= (PKSAM9)	X coordinate of aircraft in terms of coordinate system centered on impact point. The XY plane in this system is the engagement plane

-Y-

YBAR	= (JMEM6)	Average release altitude of bombs in string (feet)
YF	= (JMEM6)	Release altitude of first bomb in string (feet)
YL	= (JMEM6)	Release altitude of last bomb in string (feet)
YPOS9	= (PKSAM9)	Y coordinate of aircraft in terms of impact point coordinate system (see XPOS9)

-Z-

Z7	= (ROUTE7)	Ground range from runway to climb point
ZPOS9	= (PKSAM9)	Z coordinate of aircraft in terms of impact point coordinate system (see XPOS9)

APPENDIX B

SLAM COMPUTER MODEL

```
gen,counterair,sample,02/11/84,1,,,,,,72; sample of slam cards
initialize,0.0,250.0;
intlc,xx(36)=10.0,xx(37)=17.0; threat tr/aq times
intlc,xx(61)=1.0,xx(62)=1.0;   a/c 1 and 2 awareness flags
intlc,xx(70)=1.0;              psee factor
intlc,xx(79)=2.0;              time to impact freeze
intlc,xx(80)=0.0,xx(81)=0.0;   a/c 1 and 2 break pK  acft(i,54)
intlc,xx(82)=2.0;              ac1 weapon type       acft(1,1)
intlc,xx(83)=0.0;              ac1 dive              acft(1,2)
intlc,xx(84)=200.0;            ac1 release alt       acft(1,4)
intlc,xx(85)=0.135;            ac1 intervolometer    acft(1,6)
intlc,xx(86)=250.0;            ac1 cep               acft(1,7)
intlc,xx(87)=2.0;              ac2 weapon type       acft(2,1)
intlc,xx(88)=-10.0;            ac2 dive angle        acft(2,2)
intlc,xx(89)=600.0;            ac2 release alt       acft(2,4)
intlc,xx(90)=0.135;            ac2 intervolometer    acft(2,6)
intlc,xx(91)=125.0;            ac2 cep               acft(2,7)
intlc,xx(100)=1.0;             jam option(0=no,1=yes)
limits,0,0,100;
cont,0,25,,0.05,0.5,w,0,0.000005;
sevnt,3,ss(22),xn,200.0,200.0; compute head/pitch to next nav pt ac1
sevnt,4,ss(23),xn,200.0,200.0; compute head/pitch to next nav pt ac2
sevnt,5,ss(4),x,xx(18),0.0;    set heading flag ac1 to zero
sevnt,6,ss(9),x,xx(28),0.0;    set heading flag ac2 to zero
sevnt,7,ss(5),x,xx(19),0.0;    set pitch flag ac1 to zero
sevnt,8,ss(10),x,xx(29),0.0;   set pitch flag ac2 to zero
sevnt,9,ss(21),xp,ss(24),0.0;  getout after ac1 045 turn off tgt
sevnt,10,ss(21),xp,ss(25),0.0; getout after ac2 045 turn off tgt
sevnt,17,ss(21),xp,xx(38),0.0; schdl launch availibility for site 1
sevnt,18,ss(21),xp,xx(39),0.0; schdl launch availability for site 2
sevnt,31,ss(17),xn,9807.0,0.0; schdl aaa fire site 3 for ac1in range
sevnt,32,ss(18),xn,9807.0,0.0; schdl aaa fire site 3 for ac2in range
sevnt,33,ss(19),xn,9807.0,0.0; schdl aaa fire site 4 for ac1in range
sevnt,34,ss(20),xn,9807.0,0.0; schdl aaa fire site 4 for ac2in range
sevnt,19,ss(21),xp,xx(40),0.0; call pKaaa site 3
sevnt,20,ss(21),xp,xx(41),0.0; call pKaaa site 4
sevnt,21,ss(17),xp,9807.0,0.0; stops aaa 3 tracking ac1 (out of range)
sevnt,21,ss(18),xp,9807.0,0.0; stops aaa 3 tracking ac2 (out of range)
sevnt,22,ss(19),xp,9807.0,0.0; stops aaa 4 tracking ac1 (out of range)
sevnt,22,ss(20),xp,9807.0,0.0; stops aaa 4 tracking ac2 (out of range)
sevnt,23,ss(21),xp,xx(55),0.0; impact location update missle 1
sevnt,24,ss(21),xp,xx(56),0.0; impact location update missle 2
sevnt,25,ss(21),xp,xx(57),0.0; impact location update missle 3
sevnt,26,ss(21),xp,xx(58),0.0; impact location update missle 4
fin;
```

```
      program main

c     the following is a listing of slam variables and major
c     arrays used in this program

c     slam global variables:
c       ss(1) =ac1 present x position
c       ss(2) =ac1 present y position
c       ss(3) =ac1 present z position
c       ss(4) =ac1 present heading
c       ss(5) =ac1 present pitch
c       ss(6) =ac2 present x position
c       ss(7) =ac2 present y position
c       ss(8) =ac2 present z position
c       ss(9) =ac2 present heading
c       ss(10)=ac2 present pitch
c       ss(11)=ac1 ground range to runway center
c       ss(12)=ac2 ground range to runway center
c       ss(13)=ac1 ground range to sam site 1
c       ss(14)=ac2 ground range to sam site 1
c       ss(15)=ac1 ground range to sam site 2
c       ss(16)=ac2 ground range to sam site 2
c       ss(17)=ac1 ground range to aaa site 3
c       ss(18)=ac2 ground range to aaa site 3
c       ss(19)=ac1 ground range to aaa site 4
c       ss(20)=ac2 ground range to aaa site 4
c       ss(21)=tnow(present time)
c       ss(22)=ac1 ground range to next nav point
c       ss(23)=ac2 ground range to next nav point
c     . ss(24)=time ac1 will complete post bomb release maneuver
c       ss(25)=time ac2 will complete post bomb release maneuver

c     slam global variables:
c            xx(1)=ac1 speed(ft/sec)
c            xx(2)=ac2 speed(ft/sec)
c            xx(3)=sam site 1 x position
c            xx(4)=sam site 1 y position
c            xx(5)=sam site 1 z position
c            xx(6)=sam site 2 x position
c            xx(7)=sam site 2 y position
c            xx(8)=sam site 2 z position
c            xx(9)=aaa site 3 x position
c           xx(10)=aaa site 3 y position
c           xx(11)=aaa site 3 z position
c           xx(12)=aaa site 4 x position
c           xx(13)=aaa site 4 y position
c           xx(14)=aaa site 4 z position
c           xx(15)=ac1 next nav pt x position
c           xx(16)=ac1 next nav pt y position
c           xx(17)=ac1 next nav pt z position
```

```
c         xx(18)=ac1 next desired heading
c         xx(19)=ac1 next desired pitch
c         xx(20)=angle theta in x,y plane with ac1 heading
c         xx(21)=angle theta in x,y plane with ac2 heading
c         xx(22)=angle phi in x,z plane with ac1 pitch
c         xx(23)=angle phi in x,z plane with ac2 pitch
c         xx(24)=
c         xx(25)=ac2 next nav pt x position
c         xx(26)=ac2 next nav pt y position
c         xx(27)=ac2 next nav pt z position
c         xx(28)=ac2 next desired heading
c         xx(29)=ac2 next desired pitch
c         xx(30)=time for ac1 to depart entry point
c         xx(31)=time for ac2 to depart entry point
c         xx(32)=max range(ft) site 1
c         xx(33)=max range(ft) site 2
c         xx(34)=max range(ft) site 3
c         xx(35)=max range(ft) site 4
c         xx(36)=track and aquisition time site 1 (th(1,11))
c         xx(37)=track and aquisition time site 2 (th(2,11))
c         xx(38)=ready fire time site 1
c         xx(39)=ready fire time site 2
c         xx(40)=ready fire time site 3
c         xx(41)=ready fire time site 4
c         xx(42)=missile 1 launch time
c         xx(43)=missile 2 launch time
c         xx(44)=missile 3 launch time
c         xx(45)=missile 4 launch time
c         xx(46)=missile 1 destruct time
c         xx(47)=missile 2 destruct time
c         xx(48)=missile 3 destruct time
c         xx(49)=missile 4 destruct time
c         xx(50)=
c         xx(51)=missile 1 impact time
c         xx(52)=missile 2 impact time
c         xx(53)=missile 3 impact time
c         xx(54)=missile 4 impact time
c         xx(55)=next impact calculation time missile 1
c         xx(56)=next impact calculation time missile 2
c         xx(57)=next impact calculation time missile 3
c         xx(58)=next impact calculation time missile 4
c         xx(59)=ac1 next possible aware time for nearest missile
c         xx(60)=ac2 next possible aware time for nearest missile
c         xx(61)=ac1 aware flag(1=aware,2=not aware)
c         xx(62)=ac2 aware flag(1=aware,2=not aware)
c         xx(63)=
c         xx(64)=ac1 turn flag
c         xx(65)=ac1 pitch flag
c         xx(66)=ac2 turn flag
c         xx(67)=ac2 pitch flag
c         xx(68)=
c          xx(69)=
c          xx(70)=
c          xx(71)=missile 1 update time
c          xx(72)=missile 2 update time
c          xx(73)=missile 3 update time
c          xx(74)=missile 4 update time
c          xx(75)=missile 1 time step
c          xx(76)=missile 2 time step
c          xx(77)=missile 3 time step
c          xx(78)=missile 4 time step
c          xx(79)=time to impact freeze time
c          xx(80)=ac1 break pk (acft(1,54))
c          xx(81)=ac2 break pk (acft(2,54))
c          xx(82)=ac1 weapon type(1=high drag,2=low drag)
c          xx(83)=ac1 dive angle(0,-10 or 0,-15)
c          xx(84)=ac1 release alt(200,600 or 500,1750)
c          xx(85)=ac1 intervolometer setting(.135,.135 or .065,.156)
c          xx(86)=ac1 cep(250,125 or 250,125)
c          xx(87)=ac2 weapon type
c          xx(88)=ac2 dive angle
c          xx(89)=ac2 release alt
c          xx(90)=ac2 intervolometer setting
c          xx(91)=ac2 cep
c          xx(92)=ac1 termination at 61000 ft
c          xx(93)=ac2 termination at 61000 ft
c          xx(94)=flag:ac1 at 8000 ft to next nav point
c          xx(95)=flag:ac2 at 8000 ft to next nav point
c          xx(96)=flag:ac1 above 100 ft
c          xx(97)=flag:ac2 above 100 ft
c          xx(98)=flag:ac1 less than or equal 50 ft
c          xx(99)=flag:ac2 less than or equal 50 ft
c         xx(100)=jam option(1=on,0=off)

c      array acft(i,j) meanings:
c        i=tail numbers: 1=lead,2=wingman
c        j meanings:
c        1=weapon type: 1=ld, 2=hd
c        2=dive angle (degrees)
c        3=release airspeed (knots)
c        4=release altitude last weapon (feet)
c        5=bombs released per release pulse (integer)
c        6=intervolometer setting between pulses (seconds)
c        7=circular error probable ground plane (feet)
c        8=weapon reliability
c        9=total number of weapons on aircraft
c       10=width between aircraft weapon stations
c       11=a/c break flag(1=a/c break)
c       12=release altitude first weapon (feet)
c       13=ground range for weapon one-release point to impact (feet)
c       14=ground range to target from release point for weapon one
c       15=runin heading (on final) (degrees)
```

```
c       16=rollin direction (if pop)
c       17=ingress direction (heading in degrees)
c       18=ingress altitude(feet)
c       19
c       20
c       21=release point last weapon x coordinates
c       22=release point last weapon y coordinates
c       23=release point last weapon z coordinates
c       24=heading into last release point
c       25=pitch into last release point
c       26=release point 1st weapon x coordinate
c       27=release point 1st weapon y coordinate
c       28=release point 1st weapon z coordinate
c       29=heading into first release point
c       30=pitch into first release point
c       31=track point x coordinate
c       32=track point y coordinate
c       33=track point z coordinate
c       34=heading into track point
c       35=pitch into track point
c       36=rollin point (pop) or level off point (level) x coordinate
c       37=rollin point (pop) or level off point (level) y coordinate
c       38=rollin point (pop) or level off point (level) z coordinate
c       39=heading into rollin/level off point
c       40=pitch into rollin/level off point
c       41=pullup point (pop) or climb point (level) x coordinate
c       42=pullup point (pop) or climb point (level) y coordinate
c       43=pullup point (pop) or climb point (level) z coordinate
c       44=heading into pullup/climb point
c       45=pitch into pullup/climb point
c       46=entry point x coordinate
c       47=entry point y coordinate
c       48=entry point z coordinate
c       49=heading into entry point
c       50=pitch into entry point
c       51=egress turn off target (1=right,2=left)
c       52=total distance to target (feet)
c       53=estimated time enroute (ete) entry point to target (secs)
c       54=pk required for break
c       55=pk missile 1
c       56=pk missile 2
c       57=pk missile 3
c       58=pk missile 4
c       59=pk aaa3
c       60=pk aaa4
c       61=probability of arrival for bomb release
c       62=probability of acft survival through exit
c       63=probability of seeing target at release point
c       64=probability of jmem damage
c       65=probability of target damage per aircraft
c       66=
c        67=
c        68=
c        69=
c        70=

c      array tgt(i) meanings:
c         1=target center x coordinate
c         2=target center y coordinate
c         3=target center z coordinate
c         4=centerline (degrees)
c         5=target length (feet)
c         6=target width (feet)

c      array th(i,j) meanings:
c         i=threat site number
c          1=sam site 1
c          2=sam site 2
c          3=aaa site 3
c          4=aaa site 4
c         j meanings
c          1 =threat site x coordinate
c          2 =threat site y coordinate
c          3 =threat site z coordinate
c          4 =minimum engagement range(feet)
c          5 =maximum engagement range(feet)
c          6 =ammo supply(2=2missiles,1=1missile,0=0missiles)
c          7 =engagement status(2=firing,1=tracking,0=not engaged)
c          8 =confound delay completion time
c          9 =
c         10 =tail number of a/c site is tracking(1=lead,2=wingman)
c         11 =site track and aquisition time
c         12 =site missile velocity(knots)
c         13 =missile max time of flight at launch

c         array msl(i,j) meanings:
c         i meanings
c          1=missile 1 site 1
c          2=missile 2 site 1
c          3=missile 1 site 2
c          4=missile 2 site 2
c         j meanings
c            1=launch site
c            2=target aircraft
c            3=current impact time
c            4=self destruct based on max tof available
c            5=pkill of tgt function of impact pt and launch site
c            6=current(last calculated) missile x position
c            7=current(last calculated) missile y position
c            8=current(last calculated) missile z position
c            9=current(last calculated) missile heading
c           10=current(last calculated) missile pitch
```

```
c        11=impact point x coordinate
c        12=impact point y coordinate
c        13=impact point z coordinate
c        14=missile velocity(Knots)
c        15=missile fire time
c        16=slant range (missile position to impact point)
c        17=time to impact
c        18=
c        19=
c        20=

      dimension nset(1000)

      common/scom1/atrib(100),dd(100),
     *   ddl(100),dtnow,ii,mfa,
     *   mstop,nclnr,ncrdr,nprnt,
     *   nnrun,nnset,ntape,ss(100),
     *   ssl(100),tnext,tnow,xx(100)
        common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
        common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,
     *        by2,bz2,hdg2,mb2,gr2,aa2,sr2
        common/foley3/fnlld4,fnlwg4,popri4,sarc
        common/gress1/head1,head2,pitch1,pitch2,Kntr1,Kntr2,
     *        g1,g2

        common/gress2/ acftx8,acfty8,acftz8,velx8,vely8,velz8,
     *   v8,tofmx8,grimp8,mslx8,msly8,mslz8,pch8,hdg8,
     *   sr8,flag8,samtp9,sr9,aa9,pK9,sr10,av10,tgtg10,pK10,
     *   imptx8,impty8,imptz8,tti8,jam9,imsl3,r9,boom9

      common qset(1000)
      equivalence (nset(1),qset(1))
      nnset=1000
      ncrdr=5
      nprnt=6
      ntape=7
      call slam
      stop
      end



      subroutine state
      common/scom1/atrib(100),dd(100),
     *   ddl(100),dtnow,ii,mfa,
     *   mstop,nclnr,ncrdr,nprnt,
     *   nnrun,nnset,ntape,ss(100),
     *   ssl(100),tnext,tnow,xx(100)
        common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
        common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,
     *        by2,bz2,hdg2,mb2,gr2,aa2,sr2
```

```
        common/foley3/fnlld4,fnlwg4,popri4,sarc
        common/gress1/head1,head2,pitch1,pitch2,kntr1,kntr2,
     *        g1,g2

        common/gress2/ acftx8,acfty8,acftz8,velx8,vely8,velz8,
     *  v8,tofmx8,grimp8,mslx8,msly8,mslz8,pch8,hdg8,
     *  sr8,flag8,samtp9,sr9,aa9,pk9,sr10,av10,tgtg10,pk10,
     *  imptx8,impty8,imptz8,tti8,jam9,imsl3,r9,boom9


c     ac1 heading and pitch rates
c     xx(64)=ac1 turn flag, xx(65)=ac1 pitch flag

        if (head1 .gt. 0.5) then
      ss(4)=ssl(4) + dtnow*xx(64)*g1*32.2*57.3/xx(1)
        if (ss(4) .gt. 360.0) ss(4)=ss(4) - 360.0
        if (ss(4) .lt. 0.0) ss(4)=360.0 + ss(4)
        else
      ss(4)=ssl(4) + dtnow*0.0
        endif

        if (pitch1 .gt. 0.5) then
           ss(5)=ssl(5) + dtnow*xx(65)*8.3
        else
           ss(5)=ssl(5) + dtnow*0.0
        endif


c       angle in x,y plane for ac1

        xx(20)=360.0-ss(4)+90.0
        if (xx(20) .gt. 360.0) xx(20)=xx(20)-360.0

c       angle in z plane ac1

        xx(22)=90.0-ss(5)

c     a/c 1 (xyz) position rates

c       xrate=velocity*sin(phi/57.3)*cos(theta/57.3)
c       yrate=velocity*sin(phi/57.3)*sin(theta/57.3)
c       zrate=velocity*cos(phi/57.3)

      ss(1)=ssl(1) + dtnow*xx(1)*sin(xx(22)/57.3)*cos(xx(20)/57.3)
      ss(2)=ssl(2) + dtnow*xx(1)*sin(xx(22)/57.3)*sin(xx(20)/57.3)
      ss(3)=ssl(3) + dtnow*xx(1)*cos(xx(22)/57.3)


c       ac2 heading and pitch rates
c       xx(66)=ac2 turn flag   xx(67)=ac2 pitch flag
```

```
        if (head2 .gt. 0.5) then
      ss(9)=ssl(9) + dtnow*xx(66)*g2*32.2*57.3/xx(2)
        if (ss(9) .gt. 360.0) ss(9)=ss(9) - 360.0
        if (ss(9) .lt. 0.0) ss(9)=360.0 + ss(9)
        else
      ss(9)=ssl(9) + dtnow*0.0
        endif

        if (pitch2 .gt. 0.5) then
           ss(10)=ssl(10) + dtnow*xx(67)*8.3
        else
           ss(10)=ssl(10) + dtnow*0.0
        endif

c       angle in x,y plane ac2

        xx(21)=360.0-ss(9)+90.0
        if (xx(21) .gt. 360.) xx(21)=xx(21)-360.0

c       angle in z plane ac2

        xx(23)=90.0-ss(10)

c     ac2 position,heading,and pitch rates
      ss(6)=ssl(6) + dtnow*xx(2)*sin(xx(23)/57.3)*cos(xx(21)/57.3)
      ss(7)=ssl(7) + dtnow*xx(2)*sin(xx(23)/57.3)*sin(xx(21)/57.3)
      ss(8)=ssl(8) + dtnow*xx(2)*cos(xx(23)/57.3)


c     this computes gr for both ac from present pos to new point
c     state=(ac x-pos)-(site x-pos)**2 +(ac y-pos)-(site y-pos)**2

c     gnd range(gr) to rwy ctr ac1 and ac2 respectively

      ss(11)=sqrt(ss(1)*ss(1) + ss(2)*ss(2))
      ss(12)=sqrt(ss(6)*ss(6) + ss(7)*ss(7))

c     gr to threat site 1

      ss(13)=sqrt((ss(1)-xx(3))**2 + (ss(2)-xx(4))**2)
      ss(14)=sqrt((ss(6)-xx(3))**2 + (ss(7)-xx(4))**2)

c     gr to threat site 2

      ss(15)=sqrt((ss(1)-xx(6))**2 + (ss(2)-xx(7))**2)
      ss(16)=sqrt((ss(6)-xx(6))**2 + (ss(7)-xx(7))**2)

c     gr to threat site 3

      ss(17)=sqrt((ss(1)-xx(9))**2 + (ss(2)-xx(10))**2)
      ss(18)=sqrt((ss(6)-xx(9))**2 + (ss(7)-xx(10))**2)
```

```
c     gr to threat site 4

      ss(19)=sqrt((ss(1)-xx(12))**2 + (ss(2)-xx(13))**2)
      ss(20)=sqrt((ss(6)-xx(12))**2 + (ss(7)-xx(13))**2)

c     ss(21)=tnow

      ss(21)=ssl(21) + dtnow * 1

c     gr to next nav point
c     xx(15)=ac1 next x-coord, xx(16)=ac1 next y-coord
c     xx(25)=ac2 next x-coord, xx(26)=ac2 next y-coord

      ss(22)=sqrt((ss(1)-xx(15))**2 + (ss(2)-xx(16))**2)
      ss(23)=sqrt((ss(6)-xx(25))**2 + (ss(7)-xx(26))**2)

c       schedule a/c termination at 61000ft
        if (ss(11) .ge. xx(92)) then
           call schdl(13,0.000001,atrib)
           xx(92)=200000.0
        endif

        if (ss(12) .ge. xx(93)) then
           call schdl(14,0.000001,atrib)
           xx(93)=200000.0
        endif

c       schedule nav correction at 8000ft from first nav pt
        if (ss(22) .le. xx(94)) then
           call schdl(3,0.000001,atrib)
           xx(94)=-1000.0
        endif

        if (ss(23) .le. xx(95)) then
           call schdl(4,0.000001,atrib)
           xx(95)=-1000.0
        endif

c       schedule threat search when a/c climbs above 100ft
        if (ss(3) .ge. xx(96)) then
           call schdl(15,0.000001,atrib)
           xx(96)=100000.0
        endif

        if (ss(8) .ge. xx(97)) then
           call schdl(16,0.000001,atrib)
           xx(97)=100000.0
        endif

c       schedule a/c stop descent at 50ft off tgt
```

```
        if (ss(3) .le. xx(98)) then
           call schdl(11,0.000001,atrib)
           xx(98)=-10000.0
        endif

        if (ss(8) .le. xx(99)) then
           call schdl(12,0.000001,atrib)
           xx(99)=-10000.0
        endif

c       determine missle pK at imact time
        if (ss(21) .ge. xx(51)) then
           call schdl(27,0.000001,atrib)
           xx(51)=1000.0
        endif

        if (ss(21) .ge. xx(52)) then
           call schdl(28,0.000001,atrib)
           xx(52)=1000.0
        endif

        if (ss(21) .ge. xx(53)) then
           call schdl(29,0.000001,atrib)
           xx(53)=1000.0
        endif

        if (ss(21) .ge. xx(54)) then
           call schdl(30,0.000001,atrib)
           xx(54)=1000.0
        endif

      return
      end



      subroutine event(i)
      common/scom1/atrib(100),dd(100),
     *  ddl(100),dtnow,ii,mfa,
     *  mstop,nclnr,ncrdr,nprnt,
     *  nnrun,nnset,ntape,ss(100),
     *  ssl(100),tnext,tnow,xx(100)
        common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
        common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,
     *      by2,bz2,hdg2,mb2,gr2,aa2,sr2
        common/foley3/fnlld4,fnlwg4,popri4,sarc
        common/gress1/head1,head2,pitch1,pitch2,Kntr1,Kntr2,
     *       g1,g2
        common/gress2/ acftx8,acfty8,acftz8,velx8,vely8,velz8,
     *  v8,tofmx8,grimp8,mslx8,msly8,mslz8,pch8,hdg8,
     *  sr8,flag8,samtp9,sr9,aa9,pK9,sr10,av10,tgtg10,pk10,
     *   imptx8,impty8,imptz8,tti8,jam9,imsl3,r9,boom9

          real mb2,msl(4,20),mslx8,msly8,mslz8,imptx8,impty8,
     *        imptz8

         goto (1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,
     *        17,18,19,20,21,22,23,24,25,26,27,28,29,
     *        30,31,32,33,34),i

          if (tnow .eq. 0.0) return

 1       xx(1)=525*1.69
          print*,'now in event 1 at',ss(21)
          print*
         return
 2       xx(2)=525*1.69
          print*,'now in event 2  at',ss(21)
          print*
         return
 3        Kntr1=Kntr1-5
          print*,'now in event 3 at',ss(21)
          print*
          if (Kntr1 .lt. 21) then

 c          set ac1 breakpK to 0.0 to force break vs all missiles

            xx(80)=0.0
            acft(1,54)=xx(80)

 c          set next x,y positions to big values
            xx(15)=120000.0
            xx(16)=120000.0

            xx(17)=ss(3)
            if (acft(1,51) .eq.1) then
              xx(18)=ss(4)+45.0
              if (xx(18) .gt. 360.) xx(18)=xx(18)-360.0
            else
              xx(18)=ss(4)-45.0
              if (xx(18) .lt. 0.0) xx(18)=xx(18)+360.0
            endif
            ss(5)=0.0
            xx(19)=0.0
            ss(24)=ss(21)+(45.0/(4*32.2*57.3/xx(1)))

 c          find ac1 probability of arrival: acft(1,61)

            acft(1,61)=(1-acft(1,55))*(1-acft(1,56))*(1-acft(1,57))*
     *                 (1-acft(1,58))*(1-acft(1,59))*(1-acft(1,60))

 c          find ac1 probability of target damage
```

```
          acft(1,65)=acft(1,61)*acft(1,63)*acft(1,64)

          do 87 j=55,65
             print*,'acft(1,',j,')=',acft(1,j)
87        continue
          print*
        else
        xx(15)=acft(1,Kntr1)
        xx(16)=acft(1,Kntr1+1)
        xx(17)=acft(1,Kntr1+2)

c       compute necessary heading xx(18) into next nav point
        ax2=ss(1)
        ay2=ss(2)
        az2=ss(3)
        bx2=xx(15)
        by2=xx(16)
        bz2=xx(17)
        if ((Kntr1 .eq. 31) .and.(acft(1,2).lt.-0.5)) then
           xx(18)=acft(1,Kntr1+3)
        elseif (Kntr1 .eq. 21) then
           xx(18)=ss(4)
        else
           call mbran2
           xx(18)=mb2
        endif

c       computed necessary pitch xx(19) into next nav point

c       special condition for condition for pop pattern

            gr=sqrt((ss(1)-xx(15))**2 + (ss(2)-xx(16))**2)
            ht=xx(17)-ss(3)
            if (ht .ne. 0.0) then
              xx(19)=(atan(ht/gr))*57.3
              if((Kntr1.eq.26).and.(acft(1,2).lt.-0.5)) xx(19)=xx(19)-5.0
              if(Kntr1 .eq. 21) xx(19)=ss(5)
            else
              xx(19)=0.0
            endif
        endif

c       check if heading correction needed and set heading flag
        if (ss(4) .ne. xx(18)) then
           head1=1.0
           g1=4.0
           temp1=ss(4)-xx(18)
           if (temp1 .lt. 0.0)  temp1=360.0+temp1
           if (temp1 .lt.180.0) xx(64)=-1.0
           if (temp1 .ge.180.0) xx(64)=1.0
```

```
      endif

c     check if pitch correction needed and set pitch flag
      if (ss(5) .ne. xx(19)) then
         pitch1=1.0
         if(xx(19) .gt. ss(5)) xx(65)=1.0
         if(xx(19) .lt. ss(5)) xx(65)=-1.0
      endif
      print*,'ss(1)=',ss(1),' ss(2)=',ss(2)
      print*,'ss(3)=',ss(3),' ss(4)=',ss(4)
      print*,'ss(5)=',ss(5),' kntr1=',kntr1
      print*,'xx(64)=',xx(64),' xx(65)=',xx(65)
      print*,'xx(15)=',xx(15),' xx(16)=',xx(16)
      print*,'xx(17)=',xx(17),' xx(18)=',xx(18)
      print*,'xx(19)=',xx(19)
      print*,'head1=',head1,' pitch1=',pitch1
      print*,'ss(21)=',ss(21),' ss(24)=',ss(24)
      print*


      return

4     kntr2=kntr2-5
      print*,'now in event 4 at',ss(21)
      print*
      if (kntr2 .lt. 21) then
        xx(81)=0.0
        acft(2,54)=xx(81)
        xx(25)=120000.0
        xx(26)=120000.0
        xx(27)=ss(8)
        if (acft(2,51) .eq. 1) then
         xx(28)=ss(9)+45.0
         if (xx(28) .gt. 360.) xx(28)=xx(28)-360.0
        else
         xx(28)=ss(9)-45.0
         if (xx(28) .lt. 0.0) xx(28)=xx(28)+360.0
        endif
        ss(10)=0.0
        xx(29)=0.0
        ss(25)=ss(21)+(45./(4*32.2*57.3/xx(2)))

c       find ac2 probability of arrival:acft(2,61)

        acft(2,61)=(1-acft(2,55))*(1-acft(2,56))*(1-acft(2,57))*
     *              (1-acft(2,58))*(1-acft(2,59))*(1-acft(2,60))

c       find ac2 probability of target damage

        acft(2,65)=acft(2,61)*acft(2,63)*acft(2,64)
```

```
          do 88 j=55,65
             print*,'acft(2,',j,')=',acft(2,j)
88        continue
          print*
        else
        xx(25)=acft(2,Kntr2)
        xx(26)=acft(2,Kntr2+1)
        xx(27)=acft(2,Kntr2+2)

c       compute necessary heading xx(28) into next nav point
        ax2=ss(6)
        ay2=ss(7)
        az2=ss(8)
        bx2=xx(25)
        by2=xx(26)
        bz2=xx(27)
        if ((Kntr2 .eq.31) .and. (acft(2,2).lt.-0.5)) then
           xx(28)=acft(2,Kntr2+3)
        elseif (Kntr2 .eq. 21) then
           xx(28)=ss(9)
        else
           call mbran2
           xx(28)=mb2
        endif

c       compute necessary pitch xx(29) into next nav point
c       special condition for pop pattern

            gr=sqrt((ss(6)-xx(25))**2 + (ss(7)-xx(26))**2)
            ht=xx(27)-ss(8)
            if (ht .ne. 0.0) then
              xx(29)=(atan(ht/gr))*57.3
              if((Kntr2.eq.26).and.(acft(2,2).lt.-0.5)) xx(29)=xx(29)-5.0
              if (Kntr2 .eq. 21) xx(29)=ss(10)
            else
              xx(29)=0.0
            endif
        endif

        if (ss(9) .ne. xx(28)) then
           head2=1.0
           g2=4.0
           temp2=ss(9)-xx(28)
           if(temp2 .lt. 0.0) temp2=360.0+temp2
           if(temp2 .lt.180.0) xx(66)=-1.0
           if(temp2 .ge.180.0) xx(66)=1.0
        endif

        if (ss(10) .ne. xx(29)) then
           pitch2=1.0
           if(xx(29) .gt. ss(10)) xx(67)=1.0
```

```
            if(xx(29) .lt. ss(10)) xx(67)=-1.0
          endif

          print*,'ss(6)=',ss(6),' ss(7)=',ss(7)
          print*,'ss(8)=',ss(8),' ss(9)=',ss(9)
          print*,'ss(10)=',ss(10),' Kntr2=',Kntr2
          print*,'xx(66)=',xx(66),' xx(67)=',xx(67)
          print*,'xx(25)=',xx(25),' xx(26)=',xx(26)
          print*,'xx(27)=',xx(27),' xx(28)=',xx(28)
          print*,'xx(29)=',xx(29)
          print*,'head2=',head2,' pitch2=',pitch2
          print*,'ss(21)=',ss(21),' ss(25)=',ss(25)
          print*


        return

5         head1=0.0
          g1=1.0
          xx(64)=0.0
          print*,'now in event 5 at',ss(21)
          print*
          print*,'ss(4)=',ss(4)
          print*
          return

6         head2=0.0
          g2=1.0
          xx(66)=0.0
          print*,'now in event 6 at',ss(21)
          print*
          print*,'ss(9)=',ss(9)
          print*
          return

7         pitch1=0.0
          xx(65)=0.0
          print*,'now in event 7 at',ss(21)
          print*
          print*,'ss(5)=',ss(5)
          print*
          return

8         pitch2=0.0
          xx(67)=0.0
          print*,'now in event 8 at',ss(21)
          print*
          print*,'ss(10)=',ss(10)
          print*
          return
```

```
9       ax2=0.0
        ay2=0.0
        az2=0.0
        bx2=ss(1)
        by2=ss(2)
        bz2=ss(3)
        call mbran2
        xx(17)=50.0
        xx(18)=mb2

        gr=9000.0
        ht=xx(17)-ss(3)
        if (ht .ne. 0.0) then
           xx(19)=(atan(ht/gr))*57.3
        else
           xx(19)=0.0
        endif

        if (ss(4) .ne. xx(18))then
           head1=1.0
           g1=4.0
           temp1=ss(4)-xx(18)
           if (temp1 .lt. 0.0)  temp1=360.0+temp1
           if (temp1 .lt. 180.0) xx(64)=-1.0
           if (temp1 .ge. 180.0) xx(64)=1.0
        endif

        if (ss(5) .ne. xx(19)) then
           pitch1=1.0
           if(xx(19) .gt. ss(5))xx(65)=1.0
           if(xx(19) .lt. ss(5))xx(65)=-1.0
        endif
        print*,'now in event 9 at',ss(21)
        print*
        print*,'ss(1)=',ss(1),' ss(2)=',ss(2)
        print*,'ss(3)=',ss(3),' ss(4)=',ss(4)
        print*,'ss(5)=',ss(5),' Kntrl=',Kntrl
        print*,'xx(64)=',xx(64),' xx(65)=',xx(65)
        print*,'xx(15)=',xx(15),' xx(16)=',xx(16)
        print*,'xx(17)=',xx(17),' xx(18)=',xx(18)
        print*,'xx(19)=',xx(19)
        print*,'head1=',head1,' pitch1=',pitch1
        print*,'ss(21)=',ss(21)
        print*
        return

10      ax2=0.0
        ay2=0.0
        az2=0.0
        bx2=ss(6)
        by2=ss(7)
```

```
         bz2=ss(8)
         call mbran2
         xx(27)=50.0
         xx(28)=mb2

         gr=9000.0
         ht=xx(27)-ss(8)
         if (ht .ne. 0.0) then
            xx(29)=(atan(ht/gr))*57.3
         else
            xx(29)=0.0
         endif

         if(ss(9) .ne. xx(28)) then
           head2=1.0
           g2=4.0
           temp2=ss(9)-xx(28)
           if(temp2 .lt. 0.0) temp2=360.0+temp2
           if(temp2 .lt. 180.0)xx(66)=-1.0
           if(temp2 .ge. 180.0)xx(66)=1.0
         endif

         if (ss(10) .ne. xx(29)) then
            pitch2=1.0
            if(xx(29) .gt. ss(10)) xx(67)=1.0
            if(xx(29) .lt. ss(10)) xx(67)=-1.0
          endif

         print*,'now in event 10 at',ss(21)
         print*
         print*,'ss(6)=',ss(6),' ss(7)=',ss(7)
         print*,'ss(8)=',ss(8),' ss(9)=',ss(9)
         print*,'ss(10)=',ss(10),' kntr2=',kntr2
         print*,'xx(66)=',xx(66),' xx(67)=',xx(67)
         print*,'xx(25)=',xx(25),' xx(26)=',xx(26)
         print*,'xx(27)=',xx(27),' xx(28)=',xx(28)
         print*,'xx(29)=',xx(29)
         print*,'head2=',head2,' pitch2=',pitch2
         print*,'ss(21)=',ss(21)
         print*
         return
11       ss(5)=0.0
         pitch1=0.0
         print*,'now in event 11 at',ss(21)
         print*
         print*,'ss(3)=',ss(3)
         print*
         return

12       ss(10)=0.0
         pitch2=0.0
```

```
        print*,'now in event 12 at',ss(21)
        print*
        print*,'ss(8)=',ss(8)
        print*
        return

13      xx(1)=0.0
        if (th(1,10) .eq. 1.0) th(1,7)=0.0
        if (th(2,10) .eq. 1.0) th(2,7)=0.0
        print*,'now in event 13 at',ss(21)
        print*
        return

14      xx(2)=0.0
        if (th(1,10) .eq. 2.0) th(1,7)=0.0
        if (th(2,10) .eq. 2.0) th(2,7)=0.0
        mstop=-1
        print*,'now in event 14 at',ss(21)
        do 42 i=1,2

c       find acft i probability of surviving to exit point

        acft(i,62)=(1-acft(i,55))*(1-acft(i,56))*(1-acft(i,57))*
     *           (1-acft(i,58))*(1-acft(i,59))*(1-acft(i,60))
          do 43 j=54,65
            print*,'acft(',i,j,')=',acft(i,j)
43        continue
42      continue

c       determine probability of mission survivability ie.
c       probability that both aircraft survive mission
            .
        psmsn=acft(1,62)*acft(2,62)

c       determine probability of mission damage to target i.e.
c       probability of target damage as result of both a/c

        pdmsn=1-(1-acft(1,65))*(1-acft(2,65))
        print*
        print*,'mission probability of acft survival=',psmsn
        print*,'mission probability of target damage=',pdmsn
        print*

        return

15      do 70 i=1,2
        print*,'now in event 15 at',ss(21)
        print*

c       check to see if idle,tracking,or firing
c       if idle then set to track
```

```
c       set site ready fire times(xx(38) and xx(39))

        print*,'th(',i,',7)=',th(i,7)
c-      check engage status
          if (th(i,7) .lt. 0.5) then
c           check for sams available
            if (th(i,6) .gt. 0.5) then
c             check if confounding delay complete
              if (th(i,8) .lt. ss(21)) then
c               set status of site(i) to track
                th(i,7)=1.0
c               set tail number of tracked aircraft
                th(i,10)=1.0
c               set ready to fire time site 1=
c               tnow + track and aquisition time
                if (i .eq. 1) xx(38)=ss(21) + th(i,11)
c               set ready to  fire time site 2
                if (i .eq. 2) xx(39)=ss(21) + th(i,11)
              endif
            endif
          endif
        print*,'th(',i,',6)=',th(i,6)
        print*,'th(',i,',8)=',th(i,8)
        print*,'th(',i,',7)=',th(i,7)
        print*,'th(',i,',10)=',th(i,10)
        print*,'th(',i,',11)=',th(i,11)
        print*,'xx(38)=',xx(38),' xx(39)=',xx(39)
        print*

70      continue

        return

16      do 71 i=1,2
        print*,'now in event 16 at',ss(21)
        print*
        print*,'th(',i,',7)=',th(i,7)
          if (th(i,7) .lt. 0.5) then
            if (th(i,6) .gt. 0.5) then
              if (th(i,8) .lt. ss(21)) then
                 th(i,7)=1.0
                 th(i,10)=2.0
                 if (i .eq. 1) xx(38)=ss(21) + th(i,11)
                 if (i .eq. 2) xx(39)=ss(21) + th(i,11)
              endif
            endif
          endif
        print*,'th(',i,',6)=',th(i,6)
        print*,'th(',i,',8)=',th(i,8)
        print*,'th(',i,',7)=',th(i,7)
        print*,'th(',i,',10)=',th(i,10)
```

```
         print*,'th(',i,',11)=',th(i,11)
         print*,'xx(38)=',xx(38),' xx(39)=',xx(39)
         print*

71       continue

         return


c        17 called at ready to fire time site 1

17       ithrt=1.0
         print*,'now in event 17 at',ss(21)
         print*
         go to 80

c        18 called at ready to fire time site 2

18       ithrt=2.0
         print*,'now in event 18 at',ss(21)
         print*
         go to 80


c        set inputs to impct8 subroutine(mslx,y,z8,acftx,y,z8,v8,tofmx8)

80       continue
         if ( ithrt .eq. 1) then
c           missle position at site 1
            mslx8=th(1,1)
            msly8=th(1,2)
            mslz8=th(1,3)
         else
c           missle position at site 2
            mslx8=th(2,1)
            msly8=th(2,2)
            mslz8=th(2,3)
         endif

c        assign values based on the  a/c site is tracking

         if (th(ithrt,10) .lt. 1.5) then
            acftx8=ss(1)
            acfty8=ss(2)
            acftz8=ss(3)
            velx8=xx(1)*sin(xx(22)/57.3)*cos(xx(20)/57.3)
            vely8=xx(1)*sin(xx(22)/57.3)*sin(xx(20)/57.3)
            velz8=xx(1)*cos(xx(22)/57.3)
         else
            acftx8=ss(6)
            acfty8=ss(7)
            acftz8=ss(8)
            velx8=xx(2)*sin(xx(23)/57.3)*cos(xx(21)/57.3)
            vely8=xx(2)*sin(xx(23)/57.3)*sin(xx(21)/57.3)
            velz8=xx(2)*cos(xx(23)/57.3)
         endif

c        velocity(Knots) and missle max time of flight
         v8=th(ithrt,12)
         tofmx8=th(ithrt,13)

         print*,'mslx8=',mslx8,' msly8=',msly8,' mslz8=',mslz8
         print*,'acftx8=',acftx8,' acfty8=',acfty8
         print*,'acftz8=',acftz8,' velx8=',velx8
         print*,'vely8=',vely8,' velz8=',velz8
         print*,'v8=',v8,' tofmx8=',tofmx8
         print*

         call impct8
         if(flag8 .gt.0.5) then
c           assign a delay
            if (ithrt .eq. 1) xx(38)=ss(21) + 5.0
            if (ithrt .eq. 2) xx(39)=ss(21) + 5.0
            return
         endif

c        ground range to impact point
         grimp=grimp8

c        check if impact point is w/in min and max range of site 1

         if (ithrt .eq. 1) then
            if((grimp .ge.th(1,4)) .and. (grimp .le.th(1,5))) then
               if (th(1,6) .gt. 1.5) then
c                sam launch time for site 1 (missle 1 and 2 respectively)
                 xx(42)=1.0
                 call schdl(23,xx(42),atrib)
                 xx(38)=ss(21) + 10.0
                 if (th(1,10) .eq. th(2,10)) then
                    th(2,7)=0.0
                    th(2,10)=0.0
                    xx(39)=1000.0
                 endif
               else
                  xx(43)=1.0
                  call schdl(24,xx(43),atrib)
                  xx(38)=1000.0
                  if (th(1,10) .eq. th(2,10)) then
                     th(2,7)=0.0
                     th(2,10)=0.0
                     xx(39)=1000.0
                  endif
```

```
              endif
           else
             xx(38)=ss(21) + 5.0
           endif
           print*,'xx(42)=',xx(42),' xx(43)=',xx(43)
           print*
        else
c       check if impact point is w/in min and max range of site 2
           if((grimp .ge.th(2,4)) .and. (grimp .le.th(2,5))) then
              if (th(2,6) .gt. 1.5) then
                 xx(44)=1.0
                 call schdl(25,xx(44),atrib)
                 xx(39)=ss(21) + 10.0
                 if (th(2,10) .eq. th(1,10)) then
                    th(1,7)=0.0
                    th(1,10)=0.0
                    xx(38)=1000.0
                 endif
              else
                 xx(45)=1.0
                 call schdl(26,xx(45),atrib)
                 xx(39)=1000.0
                 if (th(2,10) .eq. th(1,10)) then
                    th(1,7)=0.0
                    th(1,10)=0.0
                    xx(38)=1000.0
                 endif
              endif
           else
              xx(39)=ss(21) + 5.0
           endif
           print*,'xx(44)=',xx(44),' xx(45)=',xx(45)
           print*
        endif

        return

c       events 19 to 22 are aaa events

19      print*,'now in event 19 at',ss(21)
        print*
        if (th(3,10) .eq. 1.0) then
           sr10=sqrt((ss(1)-xx(9))**2+(ss(2)-xx(10))**2+(ss(3)-xx(11))**2)
           tgtg10=g1
           k=1
        else
           sr10=sqrt((ss(6)-xx(9))**2+(ss(7)-xx(10))**2+(ss(8)-xx(11))**2)
           tgtg10=g2
           k=2
        endif
        av10=55.65
        call pKaa10

        acft(K,59)=pK10
        xx(40)=1000.0
        print*,'pK10=',pK10,' acft=',K
        print*
        return

20      print*,'now in event 20 at',ss(21)
        print*
        if (th(4,10) .eq. 1) then
           sr10=sqrt((ss(1)-xx(12))**2+(ss(2)-xx(13))**2+(ss(3)-xx(14))**2)
           tgtg10=g1
           K=1
        else
           sr10=sqrt((ss(6)-xx(12))**2+(ss(7)-xx(13))**2+(ss(8)-xx(14))**2)
           tgtg10=g2
           K=2
        endif
        av10=55.65

        call pKaa10

        acft(K,60)=pK10
        xx(41)=1000.0
        print*,'pK10=',pK10,' acft=',K
        print*
        return

21      xx(40)=1000.0
        th(3,10)=0.0
        print*,'now in event 21 at',ss(21),' a/c out of range aaa3'
        print*
        return

22      xx(41)=1000.0
        th(4,10)=0.0
        print*,'now in event 22 at',ss(21),' a/c outof range aaa4'
        print*
        return

c       events 23 to 26 are missle launch conditions

23      imsl=1
        print*,'now in event 23 at',ss(21)
        print*
c       check last update time
        if (xx(71) .gt. 0.5) go to 90
c       set missle last update time to ss(21) at launch
        xx(70+imsl)=ss(21)
```

```
c       decrease number of sams available at site 1
        th(1,6)=th(1,6) - 1
c       set first missle fire time to large value
        xx(42)=1000.0
c       identify target a/c to missle
        msl(1,2)=th(1,10)
c       set missle self destruct timefor maneuvering tgt after launch
        msl(1,4)=ss(21) + th(1,13)
c       set launch time
        msl(1,15)=ss(21)

        print*,'launch time missle 1'
        print*,'th(1,6)=',th(1,6),'xx(42)=',xx(42)
        print*,'msl(1,2)=',msl(1,2),'msl(1,4)=',msl(1,4)
        print*

        go to 90

24      imsl=2
        print*,'now in event 24 at',ss(21)
        print*
        if (xx(72) .gt. 0.5) go to 90
        xx(70+imsl)=ss(21)
        th(1,6)=th(1,6) - 1
        xx(43)=1000.0
        msl(2,2)=th(1,10)
        msl(2,4)=ss(21) + th(1,13)
        msl(2,15)=ss(21)

        print*,'launch time missle 2'
        print*,'th(1,6)=',th(1,6),'xx(43)=',xx(43)
        print*,'msl(2,2)=',msl(2,2),'msl(2,4)=',msl(2,4)
        print*


        go to 90

25      imsl=3
        print*,'now in event 25 at',ss(21)
        print*
        if (xx(73) .gt. 0.5) go to 90
        xx(70+imsl)=ss(21)
        th(2,6)=th(2,6) - 1
        xx(44)=1000.0
        msl(3,2)=th(2,10)
        msl(3,4)=ss(21) + th(2,13)
        msl(3,15)=ss(21)

        print*,'launch time missle 3'
        print*,'th(2,6)=',th(2,6),'xx(44)=',xx(44)
        print*,'msl(3,2)=',msl(3,2),'msl(3,4)=',msl(3,4)
```

```
          print*


          go to 90

26        imsl=4
          print*,'now in event 26 at',ss(21)
          print*
          if (xx(74) .gt. 0.5) go to 90
          xx(70+imsl)=ss(21)
          th(2,6)=th(2,6) - 1
          xx(45)=1000.0
          msl(4,2)=th(2,10)
          msl(4,4)=ss(21) + th(2,13)
          msl(4,15)=ss(21)

          print*,'launch time missle 4'
          print*,'th(2,6)=',th(2,6),'xx(45)=',xx(45)
          print*,'msl(4,2)=',msl(4,2),'msl(4,4)=',msl(4,4)
          print*


          go to 90
c         update impact location
90        continue

          print*,'imsl=',imsl
          print*

c         compute time from last missle update
          xx(74+imsl)=ss(21)-xx(70+imsl)
          xx(70+imsl)=ss(21)

c         compute theta and phi for missle

          msl(imsl,19)=360.0-msl(imsl,9)+90.0
          if (msl(imsl,19) .gt. 360.0) msl(imsl,19)=msl(imsl,19)-360.0

          msl(imsl,18)=90.0-msl(imsl,10)

c         determine missle x,y,z locations

          msl(imsl,6)=msl(imsl,6)+msl(imsl,14)*1.689*xx(74+imsl)*
     *          sin(msl(imsl,18)/57.3)*cos(msl(imsl,19)/57.3)
          msl(imsl,7)=msl(imsl,7)+msl(imsl,14)*1.689*xx(74+imsl)*
     *          sin(msl(imsl,18)/57.3)*sin(msl(imsl,19)/57.3)
          msl(imsl,8)=msl(imsl,8)+msl(imsl,14)*1.689*xx(74+imsl)*
     *          cos(msl(imsl,18)/57.3)

c         set inputs to impact8 subroutine(mslx,y,z8,acftx,y,z8,v8,tofmx8)
```

```
      print*


      go to 90

26    imsl=4
      print*,'now in event 26 at',ss(21)
      print*
      if (xx(74) .gt. 0.5) go to 90
      xx(70+imsl)=ss(21)
      th(2,6)=th(2,6) - 1
      xx(45)=1000.0
      msl(4,2)=th(2,10)
      msl(4,4)=ss(21) + th(2,13)
      msl(4,15)=ss(21)

      print*,'launch time missle 4'
      print*,'th(2,6)=',th(2,6),'xx(45)=',xx(45)
      print*,'msl(4,2)=',msl(4,2),'msl(4,4)=',msl(4,4)
      print*


      go to 90
c     update impact location
90    continue

      print*,'imsl=',imsl
      print*

c     compute time from last missle update
      xx(74+imsl)=ss(21)-xx(70+imsl)
      xx(70+imsl)=ss(21)

c     compute theta and phi for missle

      msl(imsl,19)=360.0-msl(imsl,9)+90.0
      if (msl(imsl,19) .gt. 360.0) msl(imsl,19)=msl(imsl,19)-360.0

      msl(imsl,18)=90.0-msl(imsl,10)

c     determine missle x,y,z locations

      msl(imsl,6)=msl(imsl,6)+msl(imsl,14)*1.689*xx(74+imsl)*
     *         sin(msl(imsl,18)/57.3)*cos(msl(imsl,19)/57.3)
      msl(imsl,7)=msl(imsl,7)+msl(imsl,14)*1.689*xx(74+imsl)*
     *         sin(msl(imsl,18)/57.3)*sin(msl(imsl,19)/57.3)
      msl(imsl,8)=msl(imsl,8)+msl(imsl,14)*1.689*xx(74+imsl)*
     *         cos(msl(imsl,18)/57.3)

c     set inputs to impact8 subroutine(mslx,y,z8,acftx,y,z8,v8,tofmx8)
```

```
            mslx8=msl(imsl,6)
            msly8=msl(imsl,7)
            mslz8=msl(imsl,8)


         if (msl(imsl,2) .lt. 1.5) then
c           ac1 location and velocity components
            acftx8=ss(1)
            acfty8=ss(2)
            acftz8=ss(3)
            velx8=xx(1)*sin(xx(22)/57.3)*cos(xx(20)/57.3)
            vely8=xx(1)*sin(xx(22)/57.3)*sin(xx(20)/57.3)
            velz8=xx(1)*cos(xx(22)/57.3)

c           taware  is temporary value for probability of no awareness
c           probability of being aware=xx(61)

            taware=1-xx(61)

            draw1=unfrm(0.001,1.0,2)

            if (draw1 .gt. taware) then
               aware=1.0
            else
               aware=0.0
            endif

            K=1
            braKpK=acft(1,54)

         else

c           ac2 location and velocity components

            acftx8=ss(6)
            acfty8=ss(7)
            acftz8=ss(8)
            velx8=xx(2)*sin(xx(23)/57.3)*cos(xx(21)/57.3)
            vely8=xx(2)*sin(xx(23)/57.3)*sin(xx(21)/57.3)
            velz8=xx(2)*cos(xx(23)/57.3)

            taware=1-xx(62)

            draw2=unfrm(0.001,1.0,3)

            if (draw2 .gt. taware) then
               aware=1.0
            else
               aware=0.0
            endif
         k=2
         brakpk=acft(2,54)

      endif

      v8=msl(imsl,14)
      tofmx8=msl(imsl,4) - ss(21)

      print*,'mslx8=',mslx8,' msly8=',msly8
      print*,'mslz8=',mslz8
      print*,'acftx8=',acftx8,' acfty8=',acfty8
      print*,'acftz8=',acftz8,' velx8=',velx8
      print*,'vely8=',vely8,' velz8=',velz8
      print*,'v8=',v8,' tofmx8=',tofmx8
      print*

      call impct8

      if (flag8 .gt. 0.5) then
         xx(50+imsl)=1000.0
         xx(54+imsl)=1000.0
         pk9=0.0
         acft(k,54+imsl)=pk9
         return
      endif

      msl(imsl,11)=imptx8
      msl(imsl,12)=impty8
      msl(imsl,13)=imptz8
      msl(imsl,9)=hdg8
      msl(imsl,10)=pch8
      msl(imsl,16)=sr8
      msl(imsl,17)=tti8 + ss(21)


c     assign impact time on the clock

      xx(50+imsl)=msl(imsl,17)

c     set inputs to pksam9(r9,sr9,jam9,aa9,samtp9)

c     determine range from launch site to impact pt

      if ((imsl.eq.1) .or. (imsl.eq.2)) then
         ax2=th(1,1)
         ay2=th(1,2)
         az2=th(1,3)
      else
         ax2=th(2,1)
         ay2=th(2,2)
         az2=th(2,3)
```

```
        endif

        bx2=msl(imsl,11)
        by2=msl(imsl,12)
        bz2=msl(imsl,13)

        call mbran2
        r9=sr2

c       determine a/c slant range from impact point

        ax2=msl(imsl,11)
        ay2=msl(imsl,12)
        az2=msl(imsl,13)
        if (msl(imsl,2) .lt. 1.5) then
           bx2=ss(1)
           by2=ss(2)
           bz2=ss(3)
        else
           bx2=ss(6)
           by2=ss(7)
           bz2=ss(8)
        endif

        hdg2=msl(imsl,9)
        call mbran2

c       set jamming option(1=yes 0=no)
        jam9=xx(100)
        aa9=aa2
        sr9=sr2
        if((imsl .eq. 1) .or. (imsl .eq. 2)) samtp9=1.0
        if((imsl .eq. 3) .or. (imsl .eq. 4)) samtp9=2.0

        call pksam9
        msl(imsl,5)=pk9

c       set the 0.1 sec update time

        temp90=xx(79)+1.0

        if (msl(imsl,5) .lt. brakpk) then
           xx(54+imsl)=ss(21) + 1.0
           if (tti8 .le. temp90) xx(54+imsl)=ss(21)+0.1
        else
           if((tti8 .le. xx(79)) .and. (aware .gt. 0.5)) then
              xx(54+imsl)=1000.0
              acft(k,11)=1.0
              imsl3=imsl
              call break3
           else
```

```
              xx(54+imsl)=ss(21)+1.0
              if (tti8 .le. temp90) xx(54+imsl)=ss(21)+0.1
           endif
        endif
        print*,'xx(54+',imsl,')=',xx(54+imsl)
        print*
        return

c       events 27 to 30 are impact events

27      n=1
        print*,'now in event 27 at',ss(21)
        print*,'target a/c=',msl(1,2)
        print*
        xx(55)=1000.0
        msl(1,17)=0.0
        go to 95

28      n=2
        print*,'now in event 28 at',ss(21)
        print*,'target a/c=',msl(2,2)
        print*
        xx(56)=1000.0
        msl(2,17)=0.0
        th(1,7)=0.0
        go to 95

29      n=3
        print*,'now in event 29 at',ss(21)
        print*,'target a/c=',msl(3,2)
        print*
        xx(57)=1000.0
        msl(3,17)=0.0
        go to 95

30      n=4
        print*,'now in event 30 at',ss(21)
        print*,'target a/c=',msl(4,2)
        print*
        xx(58)=1000.0
        msl(4,17)=0.0
        th(2,7)=0.0
        go to 95

95      continue

c       set inputs to pksam9(r9,sr9,jam9,aa9,samtp9,boom9) at impact time

c       determine range from launch site to impact point

        if ((n.eq.1) .or. (n.eq.2)) then
```

```
           ax2=th(1,1)
           ay2=th(1,2)
           az2=th(1,3)
        else
           ax2=th(2,1)
           ay2=th(2,2)
           az2=th(2,3)
        endif

        bx2=msl(n,11)
        by2=msl(n,12)
        bz2=msl(n,13)

        hdg2=msl(n,9)
        call mbran2
        r9=sr2

c       determine a/c slant range from impact point

        ax2=msl(n,11)
        ay2=msl(n,12)
        az2=msl(n,13)

        if (msl(n,2) .lt. 1.5) then
           bx2=ss(1)
           by2=ss(2)
           bz2=ss(3)
           k=1
        else
           bx2=ss(6)
           by2=ss(7)
           bz2=ss(8)
           k=2
        endif

        hdg2=msl(n,9)
        call mbran2

        jam9=xx(100)
        aa9=aa2
        sr9=sr2
        boom9=1.0

        if((n.eq.1) .or. (n.eq.2)) samtp9=1.0
        if((n.eq.3) .or. (n.eq.4)) samtp9=2.0

        call pksam9
        acft(k,54+n)=pk9

c       check if probability of arrival is zero
        if (acft(k,11) .gt. 0.5) then
```

```
c          assign escape routine
           if (K.eq.1) then
             xx(18)=ss(4)
c            stop the current break maneuver in preparation for getout
             head1=0.0
             g1=1.0
             xx(64)=0.0
             print*,'stop break actions: xx(18)=',xx(18),'ss(4)=',ss(4)
             call schdl(9,1.0,atrib)
           else
c            stop the current break maneuver in preparation for getout
             xx(28)=ss(9)
             head2=0.0
             g2=1.0
             xx(66)=0.0
             print*,'stop break actions: xx(28)=',xx(28),'ss(9)=',ss(9)
             call schdl(10,1.0,atrib)
           endif
        endif
        return

c       events 31 to 34 schdule aaa fire

31      if (xx(40) .eq. 1000.0) then
           xx(40)=ss(21) + 6.0
           th(3,10)=1.0
        endif
        print*,'now in event 31 at',ss(21)
        print*
        print*,'xx(40)=',xx(40)
        print*
        return

32      if(xx(40) .eq. 1000.0) then
           xx(40)=ss(21) + 6.0
           th(3,10)=2.0
        endif
        print*,'now in event 32 at',ss(21)
        print*
        print*,'xx(40)=',xx(40)
        print*
        return

33      if (xx(41) .eq. 1000.0) then
           xx(41)=ss(21) + 6.0
           th(4,10)=1.0
        endif
        print*,'now in event 33 at',ss(21)
        print*
        print*,'xx(41)=',xx(41)
        print*
```

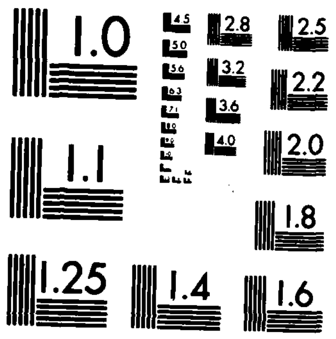
```
          return

34        if (xx(41) .eq. 1000.0) then
             xx(41)=ss(21) + 6.0
             th(4,10)=2.0
          endif
          print*,'now in event 34 at',ss(21)
          print*
          print*,'xx(41)=',xx(41)
          print*
          return

          end



       subroutine intlc
       common/scom1/atrib(100),dd(100),
     *   dd1(100),dtnow,ii,mfa,
     *   mstop,nclnr,ncrdr,nprnt,
     *   nnrun,nnset,ntape,ss(100),
     *   ssl(100),tnext,tnow,xx(100)
          common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
          common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,
     *         by2,bz2,hdg2,mb2,gr2,aa2,sr2
          common/foley3/fnlld4,fnlwg4,popri4,sarc
          common/gress1/head1,head2,pitch1,pitch2,Kntr1,Kntr2,
     *         g1,g2

          common/gress2/ acftx8,acfty8,acftz8,velx8,vely8,velz8,
     *    v8,tofmx8,grimp8,mslx8,msly8,mslz8,pch8,hdg8,
     *    sr8,flag8,samtp9,sr9,aa9,pK9,sr10,av10,tgtg10,pK10,
     *    imptx8,impty8,imptz8,tti8,jam9,imsl3,r9,boom9

          real msl(4,20)


          call bigpic

c      ac1 (xyz)  position,heading,pitch

       ss(1)=acft(1,46)
       ss(2)=acft(1,47)
       ss(3)=acft(1,48)
       ss(4)=acft(1,49)
       ss(5)=acft(1,50)

c      ac2 (xyz) position,heading,pitch

       ss(6)=acft(2,46)
       ss(7)=acft(2,47)
```

```
      ss(8)=acft(2,48)
      ss(9)=acft(2,49)
      ss(10)=acft(2,50)

c     ac velocities respectively in ft/sec

      xx(1)=0.0
      xx(2)=0.0

c     threat 1 (sam1) x y z positions

      xx(3)=th(1,1)
      xx(4)=th(1,2)
      xx(5)=th(1,3)

c     threat 2 (sam2) x y z positions

      xx(6)=th(2,1)
      xx(7)=th(2,2)
      xx(8)=th(2,3)

c     threat 3 (aaa1) x y z positions

      xx(9)=th(3,1)
      xx(10)=th(3,2)
      xx(11)=th(3,3)

c     threat 4 (aaa2) x y z positions

      xx(12)=th(4,1)
      xx(13)=th(4,2)
      xx(14)=th(4,3)

c     ac1 next nav pt (xyz-pos),next heading and pitch

      xx(15)=acft(1,41)
      xx(16)=acft(1,42)
      xx(17)=acft(1,43)
      xx(18)=acft(1,44)
      xx(19)=acft(1,45)

c     ac2 next nav pt (xyz-pos),next heading and pitch

      xx(25)=acft(2,41)
      xx(26)=acft(2,42)
      xx(27)=acft(2,43)
      xx(28)=acft(2,44)
      xx(29)=acft(2,45)

c     angle (theta) in x,y plane for ac1,ac2 respectively
```

```
      xx(20)=360.0-ss(4)+90.0
      if (xx(20) .gt. 360.) xx(20)=xx(20)-360.0
      xx(21)=360.0-ss(9)+90.0
      if (xx(21) .gt. 360.) xx(21)=xx(21)-360.0

c     angle(phi) in z plane for ac1,ac2 respectively

      xx(22)=90.0-ss(5)
      xx(23)=90.0-ss(10)

c     site1/sam1 (xyz-pos),heading,pitch

      msl(1,6)=th(1,1)
      msl(1,7)=th(1,2)
      msl(1,8)=th(1,3)
      msl(1,9)=0.0
      msl(1,10)=0.0

c     site1/sam2 (xyz-pos),heading,pitch
      msl(2,6)=th(1,1)
      msl(2,7)=th(1,2)
      msl(2,8)=th(1,3)
      msl(2,9)=0.0
      msl(2,10)=0.0

c     site2/sam1 (xyz-pos),heading,pitch

      msl(3,6)=th(2,1)
      msl(3,7)=th(2,2)
      msl(3,8)=th(2,3)
      msl(3,9)=0.0
      msl(3,10)=0.0

c     site2/sam2 (xyz-pos),heading,pitch

      msl(4,6)=th(2,1)
      msl(4,7)=th(2,2)
      msl(4,8)=th(2,3)
      msl(4,9)=0.0
      msl(4,10)=0.0

c     ac1 turn and pitch flags

      xx(64)=0.0
      xx(65)=0.0

c     ac2 turn and pitch flags

      xx(66)=0.0
      xx(67)=0.0
```

AD-A141 324 A SIMULATION MODEL TO EVALUATE AIRCRAFT SURVIVABILITY AND TARGET DAMAGE D..(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. 3/3

UNCLASSIFIED M J FOLEY ET AL. MAR 84 AFIT/GST/OS/84M-10 F/G 9/2 NL


1.0
4.5
5.0
5.6
6.3
7.1
8.0
9.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

```
c       aaa engagement times

          xx(40)=1000.0
          xx(41)=1000.0

c       initialize impact times

          xx(51)=1000.0
          xx(52)=1000.0
          xx(53)=1000.0
          xx(54)=1000.0

c       initialize descent altitude off tgt

          xx(98)=50.0
          xx(99)=50.0

c       initialize threat search altitude

          xx(96)=100.0
          xx(97)=100.0

c       initialize nav mid course correction range

          xx(94)=8000.0
          xx(95)=8000.0

c       initialize termination range

          xx(92)=61000.0
          xx(93)=61000.0

c     schedule ac departures

      xx(30)=1.0
        xx(31)=xx(30)+acft(1,53)+30.0-acft(2,53)
      call schdl(1,xx(30),atrib)
      call schdl(2,xx(31),atrib)


      do 20 m=1,25
         print*,'ss(',m,')=',ss(m)
         print*
20    continue

      do 30 n=1,99
         print*,'xx(',n,')=',xx(n)
         print*
30    continue

        kntr1=46
        Kntr2=46

        g1=1.0
        g2=1.0

        head1=0.0
        head2=0.0

        pitch1=0.0
        pitch2=0.0

      return

      end



      subroutine bigpic

      common/scom1/atrib(100),dd(100),
     *   ddl(100),dtnow,ii,mfa,
     *   mstop,nclnr,ncrdr,nprnt,
     *   nnrun,nnset,ntape,ss(100),
     *   ssl(100),tnext,tnow,xx(100)

      common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
      common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,by2,bz2,
     +   hdg2,mb2,gr2,aa2,sr2
      common/foley3/fnlld4,fnlwg4,popri4,sarc

        real msl(4,20)

c     following array values
      do 10 i=1,2
c     initialize array to zero
      do 12 K=1,70
        acft(i,K)=0.0
12     continue
10    continue

c     this applies only to the tactic: hd-level-pop
      acft(1,1)=xx(82)
      acft(1,2)=xx(83)
      acft(1,3)=525.
      acft(1,4)=xx(84)
      acft(1,5)=1
      acft(1,6)=xx(85)
      acft(1,7)=xx(86)
      acft(1,8)=0.90
      acft(1,9)=8.0
      acft(1,10)=20.
      acft(1,18)=75.0
      acft(1,54)=xx(80)

      acft(2,1)=xx(87)
      acft(2,2)=xx(88)
      acft(2,3)=525.
      acft(2,4)=xx(89)
      acft(2,5)=1.
      acft(2,6)=xx(90)
      acft(2,7)=xx(91)
      acft(2,8)=.90
      acft(2,9)=8.
      acft(2,10)=20.
      acft(2,18)=75.0
      acft(2,54)=xx(81)


c     initialize array tgt

      do 13 i=1,15
      tgt(i)=0.0

13    continue
      tgt(1)=4000.0
      tgt(2)=-4000.0
      tgt(3)=0.0
      tgt(4)=045.
      tgt(5)=550.
      tgt(6)=400.

c     initialize array th
      do 31 i=1,4
      do 32 k=1,20
        th(i,k)=0.0
32    continue
31    continue

        th(1,1)=12000.
        th(1,2)=-18000.
        th(1,3)=0.
        th(1,4)=6685.0
        th(1,5)=33456.0
        th(1,6)=2.0
        th(1,7)=0.0
        th(1,8)=0.0
        th(1,10)=0.0
        th(1,11)=xx(36)
        th(1,12)=1019.0
        th(1,13)=19.4

        th(2,1)=-12000.
```

```
          th(2,2)=18000.
          th(2,4)=13369.0
          th(2,5)=72980.0
          th(2,6)=2.0
          th(2,7)=0.0
          th(2,8)=0.0
          th(2,10)=0.0
          th(2,11)=xx(37)
          th(2,12)=1163.0
          th(2,13)=37.1

          th(3,1)=6000.
          th(3,2)=0.
          th(3,4)=9807.0

          th(4,1)=-6000.
          th(4,2)=-6000.
          th(4,4)=9807.0

c         initialize missle array to zero
          do 34 i=1,4
            do 35 k=1,20
             msl(i,k)=0.0
35          continue
34        continue

          msl(1,4)=th(1,13)
          msl(2,4)=th(1,13)
          msl(3,4)=th(2,13)
          msl(4,4)=th(2,13)
          msl(1,14)=th(1,12)
        . msl(2,14)=th(1,12)
          msl(3,14)=th(2,12)
          msl(4,14)=th(2,12)
        call scan4
        call jmem6
        call route7

        end



        subroutine jmem6
        common/scom1/atrib(100),dd(100),
       *   dd1(100),dtnow,ii,mfa,
       *   mstop,nclnr,ncrdr,nprnt,
       *   nnrun,nnset,ntape,ss(100),
       *   ssl(100),tnext,tnow,xx(100)
        common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
        common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,by2,bz2,
       +   hdg2,mb2,gr2,aa2,sr2
```

```
      common/foley3/fnlld4,fnlwg4,popri4,sarc

      real intv,la,lsk,ia,ls,lb,let,lp,nstar,n6,lep
      do 12 i=1,2
      v6=acft(i,3)
      div6=acft(i,2)
      adiv6=abs(div6)

c     target data
      la=tgt(5)
      wa=tgt(6)

c     bomb ballistic error
      if (acft(i,1).lt.1.5)sigb=4.0
      if (acft(i,1).gt.1.5)sigb=12.0

c     weapon relaibility
      r6=acft(i,8)

c     total bombs=acft(i,9)
      nr=acft(i,9)/acft(i,5)
      np=acft(i,5)
      cep=acft(i,7)
      intv=acft(i,6)
      yl=acft(i,4)
      yf=yl+(nr-1)*1.688*v6*intv*sin(adiv6/57.3)
      ybar=(yf+yl)/2.0

c     obtain impact angle and slant range
c     the impact and slant range equations are linear interpretations
c     of jmem graphs. the release airspeed is assumed = 525 knots


c     check for low drag weapon
      if(acft(i,1).lt.1.5)then

c     check for level delivery
        if(adiv6.lt.5.0)then

c     level uses 0 degree dive angle
c     equations good for release altitude 400-750 feet
      ia=(ybar+730.0)/105.0
      sr6=(ybar+850.0)/0.275
      go to 11

        else
c     delivery is ld-pop

c     pop uses -15 degree dive angle
c     equations good for release altitudes 1500 to 2500 feet
      ia=(ybar+4875.0)/250.0
      sr6=(ybar+525.0)/0.45
      go to 11

            endif

      else
c     weapon is high drag

c     check for level delivery
        if(adiv6.lt.5.0)then

c     level uses 0 degree dive angle
c     equations food for release altitudes 150 to 750 feet
      ia=(ybar+116.67)/20.4
      sr6=(ybar+685.0)/0.413
      go to 11

        else


c     delivery is hd-pop

c     pop uses -10 degree dive angle
c     equations good for release altitude 500 to 1500 feet
      ia=(ybar+166.17)/31.14
      sr6=(ybar+832.5)/0.6125
      go to 11

        endif

      endif
11    continue



      print*,'impact angle= ',ia
      print*,'sr6 slant rge=',sr6
c     so now you have sr6 and ia

c     trajectory considerations
      lsk=1.688*(sin(ia/57.3-adiv6/57.3))/sin(ia/57.3)
      sb=lsk*v6*intv
      ls=sb*(nr-1)
      ws=ocft(i,10)
      gr6=sqrt(sr6**2-yf**2)
      grt=gr6+ls/2.0
      srt=sqrt(grt**2+yf**2)
      print*,'lsk= ',lsk,' sb= ',sb,' ls= ',ls
      print*,'gr6= ',gr6,' grt= ',grt,' srt= ',srt

c     using delivery reliability of 1.0
```

```
      rd=1.0
      dep=0.573*cep
      rep=dep
      print*,'rep= ',rep,' dep= ',dep
      sigbd=sigb*srt/1000.0
      sigbr=sigbd/sin(ia/57.3)
      print*,'sigbd= ',sigbd,' sigbr= ',sigbr

c     effective target dimensions; neit=1 for maeb, =2 for maef
c     this study uses only maeb and ei=3000.
      neit=1
      ei=3000.0


c     using phd=1.0
      phd=1.0
      if(neit.eq.2)a=1-cos(ia/57.3)

      if(neit.eq.1)then
        wet=sqrt(ei)
        let=wet
        aet=ei
        endif
      if(neit.eq.2)then
        let=1.128*sqrt(ei*a)
        wet=let/a
        aet=ei
      print*,'let= ',let,' wet= ',wet,' aet= ',aet
        endif


c     single weapon effective dimensions
      wb=sqrt(wet**2+8*sigbd*sigbd)
      lb=sqrt(let**2+8*sigbr*sigbr)
      ab=lb*wb

c     stick pattern dimensions
      wp=wb+ws
      lp=lb+ls
      ap=lp*wp
      print*,'wb= ',wb,' lb= ',lb,' ab= ',ab
      print*,'wp= ',wp,' lp= ',lp,' ap= ',ap

c     effective number of weapons n6 in single weapon effective area
      n6=nr*(ab/ap)
      if(n6.lt.1.0)n6=1.0

c     conditional probability of damage
      pcd6=r6*phd*(nr/n6)*(aet/ap)

c     number of weapons in pattern
```

```
      nstor=n6*np

c     probability of damage within the pattern
      pcd=1-(1-pcd6)**nstar

c     single sortie effective pattern dimensions
      wep=max(wp,wa)
      lep=max(lp,la)
      print*,'n6= ',n6,' pcd6= ',pcd6,' nstar= ',nstar,' pcd= ',pcd
      print*,'wep= ',wep,' lep= ',lep

c     calculate expected fractional coverage--range(length)
      c6=0.6745
      t6=la/rep
      p6=lep/rep
      a6=c6*(p6+t6)/(2*sqrt(2.0))
      b6=(c6*abs(p6-t6))/(2*sqrt(2.0))
      print*,'a6= ',a6,' b6= ',b6
c     call function area5 for integral value
      aarea=area5(a6)
      barea=area5(b6)
      print*,'length aarea= ',aarea,' length barea= ',barea
      gress6=.5*(exp((-1)*a6*a6)-exp((-1)*b6*b6))
      efr=(2.3658/t6)*(a6*aarea-b6*barea+gress6)

c     calculate expected fractional coverage--deflection(width)
      c6=0.6745
      t6=wa/dep
      p6=wep/dep
      a6=c6*(p6+t6)/(2*sqrt(2.0))
      b6=(c6*abs(p6-t6))/(2*sqrt(2.0))
      print*,'a6= ',a6,' b6= ',b6
c     call function area5 for integral value
      aarea=area5(a6)
      barea=area5(b6)
      print*,'aarea= ',aarea,' barea= ',barea
      gress6=.5*(exp((-1)*a6*a6)-exp((-1)*b6*b6))
      efd=(2.36858/t6)*(a6*aarea-b6*barea+gress6)
      print*,'efr= ',efr,' efd= ',efd

c     single sortie probability of damage (before visual or survival)
      sspd=rd*pcd*(ap/(lep*wep))*efr*efd
        acft(i,64)=sspd


c       calculate probability pilot sees target at first release point
c       equations are good for ground ranges of 2000ft to 7000ft

        if (grt .lt. 2000.0) then
           print*,'error:grt less than 2000ft'
           go to 29
```

```
          endif

          if (grt .le. 3000.0) then
             psee=0.00118*yf+0.290
             if (yf .ge. 600.0) psee=0.00003*yf+0.975
             go to 29
          endif

          if (grt .le. 4000.0) then
             psee=0.00105*yf+0.183
             if (yf .ge. 600.0) psee=0.00012*yf+0.862
             go to 29
          endif

          if (grt .le. 5000.0) then
             psee=0.00113*yf+0.050
             if (yf .ge. 600.0) psee=0.00017*yf+0.758
             go to 29
          endif

          if (grt .le. 6000.0) then
             psee=0.0009*yf+0.10
             if (yf .ge. 1000.0) psee=0.00009*yf+0.836
             go to 29
          endif

          if (grt .le. 7000.0) then
             psee=0.00088*yf+0.025
             if (yf .ge. 1000.0) psee=0.00009*yf+0.783
             go to 29
          endif

29        continue
          if (psee .gt. 1.0) psee=1.0
          acft(i,63)=psee*xx(70)

c       store needed data in acft array

        acft(i,12)=yf
        acft(i,13)=gr6
        acft(i,14)=grt
          acft(i,64)=sspd
        print*,'acft(i,64)= ',acft(i,64),' acft(i,12)= ',acft(i,12)
        print*,'acft(i,13)= ',acft(i,13),' acft(i,14)= ',acft(i,14)

12      continue

c       end of giant do loop for each aircraft

        return
        end
```

```
      real function area5(upper)
c     finds area under exp(-(t**2.0))
      pi=3.14159
      error=.01
      rl=0.0
      ru=upper
      fctrl=exp((-1)*rl*rl)
      fctru=exp((-1)*ru*ru)
      K5=nint((ru*abs(fctrl-fctru))/(2*error))+1
      d=ru/K5
      esum=fctrl
      do 1 i5=1,K5
      if(i5 .lt. K5)then
        fctx=2*exp((-1)*(rl+i5*d)*(rl+i5*d))
        else
          fctx=exp((-1)*(rl+i5*d)*(rl+i5*d))
          endif
      esum=esum+fctx
      tarea=d*0.5*esum
1     continue
      area5=tarea
      print*,'fctrl= ',fctrl
      print*,'fctru= ',fctru
      print*,'K5   = ',K5
      print*,'tarea = ',tarea
      return
      end



      subroutine scan4
c
c     this is ztscan; updated 21303jan
c
      common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
      common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,by2,bz2,
     +   hdg2,mb2,gr2,aa2,sr2
      common/foley3/fnlld4,fnlwg4,popri4,sarc

      real mb1,mb2

      tgtx=tgt(1)
      tgty=tgt(2)
      tgtz=tgt(3)
      cntln=tgt(4)
      nthrt=4
      ncon12=0
      ncon6=0
      do 10 i=1,nthrt
```

```
         ax2=tgtx
         ay2=tgty
         az2=tgtz
         bx2=th(i,1)
         by2=th(i,2)
         bz2=th(i,3)
      hdg2=cntln
c
c     determine aspect of threat to tgt center
c
      call mbran2

      gr4=gr2
      aa4=aa2
      if((aa4 .gt. 300.0) .or. (aa4 .lt. 060.0)) ncon12=ncon12+1
      if((aa4 .gt. 120.0) .and. (aa4 .lt. 240.0)) ncon6=ncon6+1
      print*,'ncon12=',ncon12,' ncon6=',ncon6
10    continue
c
c     compare number of threats in 12 oclock and 6 oclock cones
c
c     and establish runin direction
c
      if(ncon12 .ge. ncon6) then
         runin=cntln
      else
         runin=cntln - 180.0
         if(runin .le. 0.0) runin=360.0 + runin
      endif
      print*,'runin=',runin
c
c     compute runin headings for both aircraft
c
c     determine point 5 miles on final
      mb1=runin-180.0
      if(mb1 .lt. 0.0) mb1=360.0+mb1
      print*,'5 mile mb1=',mb1
      gr1=5*6000.0
      ax1=tgtx
      ay1=tgty

      call xycor1

c     determine aspect of runway to 5 mile point

      ax2=bx1
      ay2=by1
      az2=0.0
      bx2=0.0
      by2=0.0
      bz2=0.0
      hdg2=runin

      call mbran2

c     determine rollin direction to final
c     popri: right=1,left=2

      aa4=aa2
      if(aa4 .lt. 180.0) then
         popri=1.0
         rninld=runin-10.0
         rninwg=runin+10.0
      else
         popri=2.0
         rninld=runin+10.0
         rninwg=runin-10.0
      endif
      if(rninld .gt. 360.0) rninld=rninld-360.0
      if(rninwg .gt. 360.0) rninwg=rninwg-360.0
      print*,'popri=',popri
      print*,'rninld=',rninld,' rninwg=',rninwg
      fnlld4=rninld
      fnlwg4=rninwg
      popri4=popri
      print*,'fnlld4= ',fnlld4,' fnlwg4= ',fnlwg4,'popri4= ',popri4
      return
      end



      subroutine route7
      common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
      common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,by2,bz2,
     +   hdg2,mb2,gr2,aa2,sr2
      common/foley3/fnlld4,fnlwg4,popri4,sarc

      real mb1,mb2,mb7

      do 73 i=1,2
        do 74 K=1,70
      print18,i,K,acft(i,k)
18    format('acft(',i2,',',i2,')=',f14.4)
74    continue
73    continue
      print*,'mb1= ',mb1,'fnlld4= ',fnlld4,' fnlwg4= ',fnlwg4
      print*,'popri4= ',popri4
c     start giant do loop for each acft planning
      do 15 i=1,2

      if(i.eq.1)acft(i,15)=fnlld4
      if(i.eq.2)acft(i,15)=fnlwg4
      acft(i,16)=popri4

c    initialize totgr
      totgr=0.0
c     find coordinates of release point of first weapon
      div7=acft(i,2)
      adiv7=abs(div7)
      v7=acft(i,3)
      altin7=acft(i,18)
      mb1=acft(i,15)-180.0
      if(mb1.lt.0.0)mb1=360+mb1
      gr1=acft(i,14)
      totgr=totgr+gr1
      print*,'gr to tgt= ',gr1,' totgr= ',totgr
      ax1=tgt(1)
      ay1=tgt(2)
      call xycor1
      acft(i,26)=bx1
      acft(i,27)=by1
      acft(i,28)=acft(i,12)
      acft(i,29)=acft(i,15)
      acft(i,30)=div7

c     find coordinates of release point of last weapon
      if  (adiv7 .gt. 5.0) then
      gr1=(acft(i,12)-acft(i,4))*sin((90-adiv7)/57.3)/sin(adiv7/57.3)
      else
      gr1=v7*1.689*acft(i,6)*((acft(i,9)/acft(i,5))-1)
      endif
      mb1=acft(i,15)
      ax1=acft(i,26)
      ay1=acft(i,27)
      call xycor1
      acft(i,21)=bx1
      acft(i,22)=by1
      acft(i,23)=acft(i,4)
      acft(i,24)=acft(i,15)
      acft(i,25)=div7

c     find coordinates of track point
      mb1=acft(i,15)-180
      if(mb1.lt.0.0)mb1=360+mb1
c     using tracktime= 5.0 seconds
      trktm=5.0
      gr1=trktm*1.688*v7*cos(adiv7/57.3)
      totgr=totgr+gr1
      print*,'gr to first release pt= ',gr1,' totgr= ',totgr
      ax1=acft(i,26)
      ay1=acft(i,27)
      call xycor1
      acft(i,31)=bx1
      acft(i,32)=by1
      acft(i,33)=acft(i,28)+(trktm*1.688*v7*sin(adiv7/57.3))
      acft(i,34)=acft(i,15)
      acft(i,35)=div7

c     check for type attack. do pop delivery points first.
      if(adiv7.lt.5.0)go to 20

c     proceed with pop calculations
c     find coordinates of rollin point with 4g turn to final
      poprad=v7*1.6889*v7*1.6889/(4.0*32.2)

c     use 30 degrees to turn for 30 degree angle off pop
      degtt=30.0
      sarc=(degtt/57.3)*poprad
      totgr=totgr+sarc
      print*,'gr to track pt= ',gr1,' totgr= ',totgr
      ang=(180-degtt)/2.0
      hypot=poprad*sin(degtt/57.3)/(sin(ang/57.3))
      print*,'sarc= ',sarc,' ang= ',ang,' hypot= ',hypot

c     this hypoteneuse is range to target so still need bearing
      d7=360-180-degtt
      tang=(180.0-d7)/2.0

c     apply pop rollin direction: 1=right, 2=left
      ridir7=acft(i,16)
      runin7=acft(i,15)
      if(ridir7.gt.1.5)then
        thdg=runin7+tang
        else
        thdg=runin7-tang
        endif
      mb1=thdg-180
      if(mb1.lt.0.0)mb1=360+mb1
      gr1=hypot
      ax1=acft(i,31)
      ay1=acft(i,32)
      call xycor1
      acft(i,36)=bx1
      acft(i,37)=by1

c     compute rollin altitude: acft(i,22)
      clmang=adiv7+5.0
      if(adiv7.lt.12.5)then
        apxalt=acft(i,12)+1000.0
        else
        apxalt=2*adiv7*100.0+(acft(i,12)/2.0)
        endif
      acft(i,38)=apxalt-(60.0*clmang)
      acft(i,40)=clmang
```

```
c     find coordinates of pull up point
      gr1=(sin((90.-clmang)/57.3))*(acft(i,38)-altin7)/
     +   sin(clmang/57.3)
      print*,'gr1 for pup point= ',gr1
      totgr=totgr+gr1
      print*,'gr to rollin pt= ',gr1,' totgr= ',totgr

c     compute runin heading (prior to rollin):  assume 30 degree
c     angle off pop
      if(ridir7.gt.1.5)then
        hdgin7=runin7+30
        mb1=hdgin7-180
        if(mb1.lt.0.0)mb1=360+mb1
        else
        hdgin7=runin7-30
        mb1=hdgin7-180
        if(mb1.lt.0.0)mb1=360+mb1
        endif
      ax1=acft(i,36)
      ay1=acft(i,37)
      call xycor1
      acft(i,41)=bx1
      acft(i,42)=by1
      acft(i,43)=altin7
      acft(i,17)=hdgin7
      if(acft(i,17).lt.0.0)acft(i,17)=360.0+acft(i,17)
      acft(i,39)=hdgin7
      acft(i,44)=hdgin7
      acft(i,45)=0.0
      go to 60

20    continue
c     now for level delivery

c     find coordinates of level off point
      mb1=acft(i,15)-180
      if(mb1.lt.0.0)mb1=360+mb1
c     reach point 3 seconds prior to track point
      gr1=3.0*1.689*v7
      totgr=totgr+gr1
      ax1=acft(i,31)
      ay1=acft(i,32)
      call xycor1
      acft(i,36)=bx1
      acft(i,37)=by1
      acft(i,38)=acft(i,12)
      acft(i,39)=acft(i,15)
      acft(i,40)=10.0
      print*,'gr to  track pt= ',gr1,'totgr= ',totgr
```

```
c     find coordinates for climb point
c     this is point to transition (10 degree climb) from ingress
c     altitude to release altitude
c     assumes this change occurs at climb angle = 5. degrees
      clmang=5.
      hdgin7=acft(i,15)
      acft(i,17)=hdgin7
      mb1=hdgin7-180
      if(mb1.lt.0)mb1=360+mb1
      gr1=(acft(i,12)-acft(i,18))*sin((90-clmang)/57.3)/sin(clmang/57.3)
      totgr=totgr+gr1
      ax1=acft(i,36)
      ay1=acft(i,37)
      call xycor1
      acft(i,41)=bx1
      acft(i,42)=by1
      acft(i,43)=acft(i,18)
      acft(i,44)=hdgin7
      acft(i,45)=0.0
      print*,'gr to level off pt= ',gr1,' totgr= ',totgr

c     now compute entry point data

60    continue

c     find coordinates of entry point

      ax2=acft(i,41)
      ay2=acft(i,42)
      az2=acft(i,43)
      bx2=0.0
      by2=0.0
      bz2=0.0
      hdg2=hdgin7
      call mbran2
      mb7=mb2
      z7=gr2
      aa7=aa2
      print*,'mb7= ',mb7,' z7= ',z7,' aa7= ',aa7

      if(aa7.le.180)then
        if(aa7.eq.180)then
          f=60000.+z7
          go to 50
        else
          if(aa7.le.90)then
            alfa7=aa7
            nalfa=-1
            go to 45
          else
```

```
              alfa7=180-aa7
              nalfa=+1
              go to 45
            endif
            endif

        else
          if(aa7.eq.360)then
            f=60000.-z7
            go to 50
          else
            if(aa7.ge.270)then
              alfa7=360-aa7
              nalfa=-1
              go to 45
            else
              alfa7=aa7-180
              nalfa=+1
              go to 45
            endif
            endif
        endif

45      continue

c       calculate distance f
        c=60000.0
        b=(sin(alfa7/57.3))*z7/(sin(90./57.3))
        d=sqrt(z7**2-b**2)
        e=sqrt(c**2-b**2)
        f=e+nalfa*d
        print*,'alfa7= ',alfa7,' nalfa= ',nalfa,' b= ',b
        print*,'d= ',d,' e= ',e,' f= ',f

50      gr1=f
        totgr=totgr+gr1
        print*,'gr to pt after ep= ',gr1,' totgr= ',totgr
        mb1=hdgin7-180
        if(mb1.lt.0.0)mb1=360+mb1
        ax1=acft(i,41)
        ay1=acft(i,42)
        call xycor1
        acft(i,46)=bx1
        acft(i,47)=by1
        acft(i,48)=altin7
        acft(i,49)=hdgin7
        acft(i,50)=0.0

c       find direction of turn off target
        if(acft(i,16).lt.1.5)then
```

```
          acft(i,51)=2.0
          else
          acft(i,51)=1.0
          endif

c       compute total ground range
        acft(i,52)=totgr
c       use constant v7 to figure ete
        acft(i,53)=totgr/(v7*1.689)

c       print out array acft
        do 14 k=1,70
        print19,i,k,acft(i,k)
19      format('acft(',i2,',',i2,')=',f14.4)
14      continue

15      continue

c       end of giant do loop
c       end of route planning
        print*,'jmem popri4= ',popri4,' fnlld4= ',fnlld4
        print*,'jmem fnlwg4= ',fnlwg4

        return
        end



        subroutine xycor1
        common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
        common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,by2,bz2,
       +hdg2,mb2,gr2,aa2,sr2
        common/foley3/fnlld4,fnlwg4,popri4,sarc

        real mb1

        if((mb1 .eq. 0.0) .or. (mb1 .eq. 360.0)) then
          dx1=0.0
          dy1=gr1
          go to 5
        endif
        if(mb1 .eq. 180.0) then
          dx1=0.0
          dy1=-gr1
          go to 5
        endif
        if (mb1 .le. 90) nquad=1
        if (mb1 .gt. 90) nquad=2
        if (mb1 .gt.180) nquad=3
        if (mb1 .gt.270) nquad=4
        go to (1,2,3,4) nquad
```

```
1     alfa1=90.0-mb1
      talfa1=alfa1/57.3
      dx1=1.0*gr1*cos(talfa1)
      dy1=1.0*gr1*sin(talfa1)
      go to 5
2     alfa1=mb1-90.0
      talfa1=alfa1/57.3
      dx1=1.0*gr1*cos(talfa1)
      dy1=(-1.0)*gr1*sin(talfa1)
      go to 5
3     alfa1=270.0-mb1
      talfa1=alfa1/57.3
      dx1=(-1.0)*gr1*cos(talfa1)
      dy1=(-1.0)*gr1*sin(talfa1)
      go to 5
4     alfa1=mb1-270.0
      talfa1=alfa1/57.3
      dx1=(-1.0)*gr1*cos(talfa1)
      dy1=1.0*gr1*sin(talfa1)
      go to 5
5     continue
      bx1=ax1+dx1
      by1=ay1+dy1
      return
      end



      subroutine mbran2
      common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
      common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,by2,bz2,
     +   hdg2,mb2,gr2,aa2,sr2
      common/foley3/fnlld4,fnlwg4,popri4,sarc

      integer sturn2
      real mb2

      dx2=bx2-ax2
      dy2=by2-ay2
      dz2=bz2-az2
      if((dx2.lt.0.5).and.(dx2.gt.-0.5))then
        alfa2=90.0
        else
        alfa2=abs((atan(dy2/dx2))*57.3)
        endif
      if(dx2 .ge. 0.0) then
         if(dy2 .ge. 0.0) mb2=90.0-alfa2
         if(dy2 .lt. 0.0) mb2=90.0+alfa2
      endif
      if(dx2 .lt. 0.0) then
         if(dy2 .ge. 0.0) mb2=270+alfa2
         if(dy2 .lt. 0.0) mb2=270-alfa2
      endif
      gr2=sqrt(dx2**2+dy2**2)
      sr2=sqrt(gr2**2+dz2**2)
      aa2=mb2-hdg2
      if(aa2 .lt. 0.0) aa2=360.0+aa2
c
c     determine shortest turn: 1=right,2=left
c
      if(aa2 .le. 180.0) then
         sturn2=1
      else
         sturn2=2
      endif
      return
      end



        subroutine pKsam9

      common/scom1/atrib(100),dd(100),
     *   dd1(100),dtnow,ii,mfa,
     *   mstop,nclnr,ncrdr,nprnt,
     *   nnrun,nnset,ntape,ss(100),
     *   ss1(100),tnext,tnow,xx(100)
        common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
        common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,
     *          by2,bz2,hdg2,mb2,gr2,aa2,sr2
        common/foley3/fn11d4,fn1wg4,popri4,sarc
        common/gress1/head1,head2,pitch1,pitch2,Kntr1,Kntr2,
     *          g1,g2
        common/gress2/ acftx8,acfty8,acftz8,velx8,vely8,velz8,
     *   v8,tofmx8,grimp8,mslx8,msly8,mslz8,pch8,hdg8,
     *   sr8,flag8,samtp9,sr9,aa9,pK9,sr10,av10,tgtg10,pK10,
     *   imptx8,impty8,imptz8,tti8,jam9,ims13,r9,boom9


      real avpc(110),theta(110),cellpK(110),mag,lr,msl(4,20)

c     ave p/cep


      avpc(1)=0.0
      avpc(2)=0.2506
      avpc(3)=0.2506
      avpc(4)=0.2506
      avpc(5)=0.2506
      avpc(6)=0.2506
      avpc(7)=0.2506
      avpc(8)=0.2506
avpc(9)=0.4722
avpc(10)=0.4722
avpc(11)=0.4722
avpc(12)=0.4722
avpc(13)=0.4722
avpc(14)=0.4722
avpc(15)=0.4722
avpc(16)=0.4722
avpc(17)=0.4722
avpc(18)=0.4722
avpc(19)=0.4722
avpc(20)=0.4722
avpc(21)=0.4722
avpc(22)=0.7162
avpc(23)=0.7162
avpc(24)=0.7162
avpc(25)=0.7162
avpc(26)=0.7162
avpc(27)=0.7162
avpc(28)=0.7162
avpc(29)=0.7162
avpc(30)=0.7162
avpc(31)=0.7162
avpc(32)=0.7162
avpc(33)=0.7162
avpc(34)=0.7162
avpc(35)=0.7162
avpc(36)=0.7162
avpc(37)=0.7162
avpc(38)=0.7162
avpc(39)=0.7162
avpc(40)=0.9950
avpc(41)=0.9950
avpc(42)=0.9950
avpc(43)=0.9950
avpc(44)=0.9950
avpc(45)=0.9950
avpc(46)=0.9950
avpc(47)=0.9950
avpc(48)=0.9950
avpc(49)=0.9950
avpc(50)=0.9950
avpc(51)=0.9950
avpc(52)=0.9950
avpc(53)=0.9950
avpc(54)=0.9950
avpc(55)=0.9950
avpc(56)=0.9950
avpc(57)=0.9950
avpc(58)=0.9950
avpc(59)=0.9950
avpc(60)=0.9950
avpc(61)=1.3300
avpc(62)=1.3300
avpc(63)=1.3300
avpc(64)=1.3300
avpc(65)=1.3300
avpc(66)=1.3300
avpc(67)=1.3300
avpc(68)=1.3300
avpc(69)=1.3300
avpc(70)=1.3300
avpc(71)=1.3300
avpc(72)=1.3300
avpc(73)=1.3300
avpc(74)=1.3300
avpc(75)=1.3300
avpc(76)=1.3300
avpc(77)=1.3300
avpc(78)=1.3300
avpc(79)=1.3300
avpc(80)=1.3300
avpc(81)=1.3300
avpc(82)=1.8170
avpc(83)=1.8170
avpc(84)=1.8170
avpc(85)=1.8170
avpc(86)=1.8170
avpc(87)=1.8170
avpc(88)=1.8170
avpc(89)=1.8170
avpc(90)=1.8170
avpc(91)=1.8170
avpc(92)=1.8170
avpc(93)=1.8170
avpc(94)=1.8170
avpc(95)=1.8170
avpc(96)=1.8170
avpc(97)=1.8170
avpc(98)=2.5370
avpc(99)=2.5370
avpc(100)=2.5370
avpc(101)=2.5370
avpc(102)=2.5370
avpc(103)=2.5370
avpc(104)=2.5370
avpc(105)=2.5370
avpc(106)=2.5370
avpc(107)=2.5370
avpc(108)=2.5370
avpc(109)=2.5370
```

```
c     theta in radians

      theta(1)=0.0
      theta(2)=0.0
      theta(3)=0.8976
      theta(4)=1.7952
      theta(5)=2.6928
      theta(6)=3.5904
      theta(7)=4.4880
      theta(8)=5.3856
      theta(9)=0.2417
      theta(10)=0.7250
      theta(11)=1.2083
      theta(12)=1.6916
      theta(13)=2.1749
      theta(14)=2.6583
      theta(15)=3.1416
      theta(16)=3.6249
      theta(17)=4.1082
      theta(18)=4.5916
      theta(19)=5.0749
      theta(20)=5.5582
      theta(21)=6.0415
      theta(22)=0.1745
      theta(23)=0.5236
      theta(24)=0.8727
      theta(25)=1.2217
      theta(26)=1.5708
      theta(27)=1.9199
      theta(28)=2.2689
      theta(29)=2.6180
      theta(30)=2.9671
      theta(31)=3.3161
      theta(32)=3.6652
      theta(33)=4.0143
      theta(34)=4.3633
      theta(35)=4.7124
      theta(36)=5.0615
      theta(37)=5.4105
      theta(38)=5.7596
      theta(39)=6.1087
      theta(40)=0.0
      theta(41)=0.2992
      theta(42)=0.5984
      theta(43)=0.8976
      theta(44)=1.1968
      theta(45)=1.4960
      theta(46)=1.7952
      theta(47)=2.0944
      theta(48)=2.3936
      theta(49)=2.6928
theta(50)=2.9920
theta(51)=3.2912
theta(52)=3.5904
theta(53)=3.8896
theta(54)=4.1888
theta(55)=4.4880
theta(56)=4.7872
theta(57)=5.0864
theta(58)=5.3856
theta(59)=5.6848
theta(60)=5.9840
theta(61)=0.1496
theta(62)=0.4488
theta(63)=0.7480
theta(64)=1.0472
theta(65)=1.3464
theta(66)=1.6456
theta(67)=1.9448
theta(68)=2.2440
theta(69)=2.5432
theta(70)=2.8424
theta(71)=3.1416
theta(72)=3.4408
theta(73)=3.7400
theta(74)=4.0392
theta(75)=4.3384
theta(76)=4.6376
theta(77)=4.9368
theta(78)=5.2360
theta(79)=5.5352
theta(80)=5.8344
theta(81)=6.1336
theta(82)=0.1964
theta(83)=0.5890
theta(84)=0.9817
theta(85)=1.3744
theta(86)=1.7671
theta(87)=2.1598
theta(88)=2.5525
theta(89)=2.9452
theta(90)=3.3380
theta(91)=3.7306
theta(92)=4.1233
theta(93)=4.5160
theta(94)=4.9087
theta(95)=5.3014
theta(96)=5.6941
theta(97)=6.0868
theta(98)=0.2618
theta(99)=0.7854
theta(100)=1.3090
```

```
      theta(101)=1.8326
      theta(102)=2.3562
      theta(103)=2.8798
      theta(104)=3.4034
      theta(105)=3.9270
      theta(106)=4.4506
      theta(107)=4.9742
      theta(108)=5.4978
      theta(109)=6.0214

c     compute pk for sam

      rm=r9/3.28
      print*,'rm=',rm
      rdb=10.0*log10(rm)
c       radar cross section
        rcs=2.5
      rcsdb=10.0*log10(rcs)

      if (samtp9 .eq. 2.0) then
         erpdb=29.6
         prdb=50.0
         grdb=43.0
         a=0.00000071
         b=2200.0
         c=58.0
         d=4.03e-27
         e=1.02e-16
         f=58.0
c       lethal radius in feet
         lr=85.93
      else
         erpdb=29.6
         prdb=53.0
         grdb=41.0
         a=0.000000325
         b=1890.0
         c=25.0
         d=8.09e-27
         e=4.84e-16
         f=25.0
c       lethal radius in feet
         lr=72.16
      endif

c       claculations based on meters

        if (jam9 .eq. 1) then
           jsdb=erpdb+11.0+2*(rdb)-prdb-grdb-rcsdb
           js=10.0**(jsdb/10.0)
           cepm=sqrt(a*js*rm**2 + b*js + c)
```

```
          else
             cepm=sqrt((d*rm**6/rcs**2)+(e*rm**4/rcs**2)+(f))
          endif

c         cep changed to feet
        cepf=cepm*3.28
        print*,'jsdb=',jsdb,' js=',js,' cepf=',cepf

          if  (boom9 .eq. 0.0) then
             go to 93
          else

c         reset boom9 to zero

          boom9=0.0
c            determine aspect and range(x,y,z) of a/c from missle impact

             if (aa9 .le. 180.0) then
                phi=abs(90.0-aa9)
             else
                phi=abs(270.0-aa9)
             endif

             mag=sr9
             xpos9=mag * cos(phi/57.3)
             ypos9=0.0
             zpos9=mag * sin(phi/57.3)
             print*,'mag=',mag
             print*,'phi=',phi,' xpos9=',xpos9,' zpos9=',zpos9
             print*

             pk9=0.0

c            determine if a/c is outside lethal zone

c            if ((aa9 .gt. 120.0) .and. (aa9 .lt.240.0))
c               if (sr9 .gt. 10) go to 92
c            endif


c            if in lethal zone compute cell pk

             do 91 i=1,109
              tpcac=avpc(i)*cepf
              gzx9=tpcac*cos(theta(i))
              gzy9=tpcac*sin(theta(i))
              gzz9=0.0

c             slant range vector components
              srx9=xpos9 - gzx9
              sry9=ypos9 - gzy9
              srz9=zpos9 - gzz9
              sr9=sqrt(srx9**2 + sry9**2 + srz9**2)
              print*,'gzx9=',gzx9,' gzy9=',gzy9,' gzz9=',gzz9
              print*,'srx9=',srx9,' sry9=',sry9,' srz9=',srz9
              print*,'sr9=',sr9

              if(sr9 .le. lr) then
                cellpk(i)=1.0
              else
                cellpk(i)=0.0
              endif

              if (i .le. 97) then
                 fac=0.01
              else
                 fac=0.0025
              endif

              pk9=pk9 + cellpk(i)*fac
              print*,'cellpk(',i,')=',cellpk(i),' pk9=',pk9
              print*

91         continue
              print*,'pksam9=',pk9
              print*
           return

92         continue
           pk9=0.0
           print*,'pksam9=',pk9,' exceeded aspect angle'
           print*
           return

93        continue
           pk9=1.0-(0.5**((lr/cepf)**2.0))
           print*,'pksam9=',pk9,' equation form'
           print*
      endif
         return
      end



        subroutine impct8

      common/scom1/atrib(100),dd(100),
     *   ddl(100),dtnow,ii,mfa,
     *   mstop,nclnr,ncrdr,nprnt,
     *   nnrun,nnset,ntape,ss(100),
     *   ssl(100),tnext,tnow,xx(100)
         common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
        common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,
     *        by2,bz2,hdg2,mb2,gr2,aa2,sr2
        common/foley3/fnlld4,fnlwg4,popri4,sarc
        common/gress1/head1,head2,pitch1,pitch2,kntr1,kntr2,
     *        g1,j2
        common/gress2/ acftx8,acfty8,acftz8,velx8,vely8,velz8,
     *   v8,tofmx8,grimp8,mslx8,msly8,mslz8,pch8,hdg8,
     *   sr8,flag8,samtp9,sr9,aa9,pk9,sr10,av10,tgtg10,pk10,
     *   imptx8,impty8,imptz8,tti8,jam9,imsl3,r9,boom9


       real mslx8,msly8,mslz8,mb2,imptx8,impty8,imptz8


      pzxrf=acftx8-mslx8
      pzyrf=acfty8-msly8
      pzzrf=acftz8-mslz8
      print*,'pzxrf= ',pzxrf,' pzyrf= ',pzyrf,' pzzrf= ',pzzrf

      sr01=pzxrf
      sr02=pzyrf
      sr03=pzzrf

c     calculate magnitude of vector sr0--call it s1
      s1=sqrt(sr01**2+sr02**2+sr03**2)
      print*,'s1= ',s1
        print*

c       time to impact

81    tti8=s1/(v8*1.689)

c     check see if aircraft will outrun missile

      flag8=0.0
      if(tti8.gt.tofmx8)then
        flag8=1.0
        print*,'acft will outrun missile since tti8= ',tti8,' sec'
        print*
        return
        endif

c     calculate acft component changesfor this tti8
      dacftx=velx8*tti8
      dacfty=vely8*tti8
      dacftz=velz8*tti8

c     calculate sr1 vector for these changes in acft position
      sr11=dacftx+pzxrf
      sr12=dacfty+pzyrf
```

```
        sr13=dacftz+pzzrf
          temp8=-(mslz8-100.0)
          if (sr13 .lt. temp8) sr13=temp8

c       calculate magnitude of vector sr1--call it s2
        s2=sqrt(sr11**2+sr12**2+sr13**2)

c       check change in slant range for possible catchup
        temp8=abs(s2-s1)
        if(temp8.gt.5.0)then
           if(s1.gt.s2)then
             s1=(s1+s2)/2.0
           else
             s1=s2
           endif
           go to 81
        endif

          sr8=s2

c       post catchup calculations
        dmslx=sr11
        dmsly=sr12
        dmslz=sr13
        tti8=tti8

        imptx8=mslx8+dmslx
        impty8=msly8+dmsly
        imptz8=mslz8+dmslz

        pch8=(asin(dmslz/s2))*57.3

c       grimp8 is ground range from original missile pt to impact pt
        grimp8=sqrt((imptx8-mslx8)**2+(impty8-msly8)**2)

c         determine heading of missle to impact point

          ax2=mslx8
          ay2=msly8
          az2=mslz8
          bx2=imptx8
          by2=impty8
          bz2=imptz8
          call mbran2
          hdg8=mb2

        print*,'imptx8= ',imptx8,' impty8= ',impty8,' imptz8= ',imptz8
        print*,'pch8=',pch8,' hdg8=',hdg8
          print*,'grimp8=',grimp8,' sr8=',sr8
          print*,'tti8=',tti8
          print*
```

```
      return
    end


      subroutine breaK3

    common/scom1/atrib(100),dd(100),
 *    dd1(100),dtnow,ii,mfa,
 *    mstop,nclnr,ncrdr,nprnt,
 *    nnrun,nnset,ntape,ss(100),
 *    ss1(100),tnext,tnow,xx(100)
      common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
      common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,
 *          by2,bz2,hdg2,mb2,gr2,aa2,sr2
      common/foley3/fnlld4,fnlwg4,popri4,sarc
      common/gress1/head1,head2,pitch1,pitch2,Kntr1,Kntr2,
 *          g1,g2
      common/gress2/ acftx8,acfty8,acftz8,velx8,vely8,velz8,
 *    v8,tofmx8,grimp8,mslx8,msly8,mslz8,pch8,hdg8,
 *    sr8,flag8,samtp9,sr9,aa9,pK9,sr10,av10,tgtg10,pK10,
 *    imptx8,impty8,imptz8,tti8,jam9,imsl3,r9,boom9


      real msl(4,20)

      i=imsl3
      if (msl(i,2) .lt. 1.5) then
         ax2=ss(1)
         ay2=ss(2)
         az2=ss(3)
         hdg2=ss(4)
      else
         ax2=ss(6)
         ay2=ss(7)
         az2=ss(8)
         hdg2=ss(9)
      endif
      bx2=msl(i,6)
      by2=msl(i,7)
      bz2=msl(i,8)

      call mbran2
      aa3=aa2

      if (msl(i,2) .lt. 1.5) then
         if (aa3 .lt. 180.0) then
           xx(64)=1.0
         else
           xx(64)=-1.0
```

```
   endif

   xx(18)=msl(i,9)-180.0
      if (xx(18) .lt. 0.0) xx(18)=xx(18) + 360.0

   if((aa3 .ge.345.0).and.(aa3.le.360.0)) then
      xx(18)=xx(18)-90.0
      if (xx(18) .lt. 0.0) xx(18)=xx(18)+360.0
   elseif ((aa3.ge.0.0).and.(aa3.le.15.0)) then
      xx(18)=xx(18)+90.0
      if (xx(18) .gt. 360.0) xx(18)=xx(18)-360.0
   endif

   head1=1.0
   g1=8.0
   ss(5)=0.0
   pitch1=0.0
   xx(65)=0.0
   print*,'ax2=',ax2,' ay2=',ay2,' az2=',az2
   print*,'bx2=',bx2,' by2=',by2,' bz2=',bz2
   print*,'ac1 heading=',hdg2,' msl heading=',msl(i,9)
   print*,'ac1 break heading=',xx(18),' xx(64)=',xx(64)
   print*,'aa3=',aa3
   print*

else
   if (aa3 .lt. 180.0) then
      xx(66)=1.0
   else
      xx(66)=-1.0
   endif

   if ((aa3.ge.315.0).and.(aa3.le.360.0)) then
      xx(28)=aa3-90.0
   elseif ((aa3.ge.0.0).and.(aa3.le.45.0)) then
      xx(28)=aa3+90.0
   else
      xx(28)=msl(i,9)-180.0
      if (xx(28) .lt. 0.0) xx(28)=xx(28)+360.0
   endif

   head2=1.0
   g2=8.0
   ss(10)=0.0
   pitch2=0.0
   xx(67)=0.0
   print*,'ax2=',ax2,' ay2=',ay2,' az2=',az2
   print*,'bx2=',bx2,' by2=',by2,' bz2=',bz2
   print*,'ac2 heading =',hdg2,' msl heading=',msl(i,9)
   print*,'ac2 break heading=',xx(28),' xx(66)=',xx(66)
   print*,'aa3=',aa3
```

```
           print*
         endif

         return
         end



         subroutine pkaa10
       common/scom1/atrib(100),dd(100),
     *   dd1(100),dtnow,ii,mfa,
     *   mstop,nclnr,ncrdr,nprnt,
     *   nnrun,nnset,ntape,ss(100),
     *   ss1(100),tnext,tnow,xx(100)
         common/foley1/acft(2,70),tgt(15),th(4,20),msl(4,20)
         common/foley2/ax1,ay1,mb1,gr1,bx1,by1,ax2,ay2,az2,bx2,
     *        by2,bz2,hdg2,mb2,gr2,aa2,sr2
         common/foley3/fnlld4,fnlwg4,popri4,sarc
         common/guess1/head1,head2,pitch1,pitch2,Kntr1,Kntr2,
     *        g1,g2

         common/gress2/ acftx8,acfty8,acftz8,velx8,vely8,velz8,
     *   v8,tofmx8,grimp8,mslx8,msly8,mslz8,pch8,hdg8,
     *   sr8,flag8,samtp9,sr9,aa9,pK9,sr10,av10,tgtg10,pK10,
     *   imptx8,impty8,imptz8,tti8,jam9,imsl3,r9,boom9


c      compute pK for aaa

       vi=3050.0
       vf=3050*exp(-0.1513*sr10/1000.0)
       tof=(6601.76/vf) - 2.1645
       sig2=2*3.1416*((20.0*sr10/1000.0)**2)
       print*,'vf=',vf,' tof=',tof,' sig2=',sig2,' tgtg10=',tgtg10
       pkss10=(av10/(sig2+av10))*exp(-0.5*(((32.2*tgtg10*(tof**2))**2)/
     *      (sig2+av10)))
       print*,'pKss10=',pKss10
c
c      rounds per burst(rds)
c
       rds=100.0
       pK10=1.0-(1.0-pKss10)**rds
       print*,'pKaaa=',pK10


         return
         end
```

APPENDIX C

ROUTING LOGIC VERIFICATION

This appendix contains logic and computer code used to verify that each aircraft correctly flew its route through the target area. The primary logic which governs the aircraft's navigation through the area is subroutine EVENT, a few actions are also scheduled via subroutine STATE. In this scenario, aircraft number one performed a level attack with high drag munitions, while aircraft number two executed a pop-to-angular high drag weapons delivery. The flight planned coordinates from subroutine ROUTE7 were as follows:

	Flight Planned Coordinates		
	X	Y	Z
Aircraft Number One			
Entry Point	-45743	-38827	75
Midcourse Correction	---	---	---
Climb Point	-5070	-10350	75
Leveloff Point	-3899	-9530	200
Track Point	-1720	-8005	200
First Release Point	1909	-5463	200
Last Release Point	2595	-4983	200
Aircraft Number Two			
Entry Point	-49610	-33746	75
Midcourse Correction	---	---	---
Pullup Point	-5027	-12958	75
Rollin Point	-2421	-11743	845
Track Point	-1	-9713	1514
First Release Point	2502	-6138	745
Last Release Point	2975	-5463	600

The verification attempted to insure that the aircraft arrived close to these points and performed the approximate delivery profiles which were planned. It was not required that the computer code 'fly' each aircraft exactly on preplanned parameters nor position the aircraft at the precise planned weapons release points and parameters. Since subroutine JMEM6 calculated target damage as a function of average release parameters, the real function of moving the aircraft through the area was to assess aircraft survival versus threats.

Events 3 and 4 were the primary events used to move the aircraft through the target area. In these events, state variables ss(1) through ss(5) for aircraft number one and variables ss(6) through ss(10) for aircraft number two represent the aircraft's current x, y, and z coordinates; heading; and pitch at the time the event is executed. The global variables xx(15) through xx(19) for aircraft number one and variables xx(25) through xx(29) for aircraft number two represent the next navigation point's x, y, and z coordinates as well as the desired heading and pitch to reach that point from the aircraft's current position.

```
now in event 5 at  .000000000e+00
```

This is scheduled via subroutine INTLC. Aircraft number receives initial ingress heading as planned in subroutine ROUTE7.

```
   ss(4)=  .550000000e+02
```

```
now in event 6 at  .000000000e+00
```

Aircraft number two receives initial ingress heading.

```
   ss(9)=  .650000000e+02
```

```
now in event 1 at  1.00000000
```

This is scheduled via subroutine INTLC. Aircraft number one is correctly scheduled to start at one second into simulation time at 525 Knots velocity.

```
now in event 2  at  .292931747e+02
```

Aircraft number two is scheduled via subroutine INTLC to start at a time that equals aircraft number one's start time plus estimated time enroute (ETE) to the target plus 30 seconds fragmentation clearance time minus the ETE of aircraft number two. In this case, aircraft number one's start time of one second plus and ETE of 68.484 seconds plus 30 seconds for fragmentation totals 99.484 seconds. With an ETE of 70.187 seconds, aircraft number two therefore is scheduled for a start at 29.29 seconds.

```
now in event 3 at  .479497185e+02
```

This is a midcourse correction to steer aircraft number one to the climb point. A slight correction to the right and down is needed, so heading and pitch flags are correctly set to plus and minus respectively.

```
ss(1)= -.116198809e+05  ss(2)= -.149353945e+05
ss(3)=  .798213882e+02  ss(4)=  .550000000e+02
ss(5)=  .000000000e+00  Kntrl= 41
xx(64)=  1.00000000  xx(65)= -.100000000e+01
xx(15)= -.507014697e+04  xx(16)= -.103503770e+05
xx(17)=  .750000000e+02  xx(18)=  .550042076e+02
xx(19)= -.345544145e-01
head1=  1.00000000  pitch1=  1.00000000
```

```
now in event 5 at  .479502258e+02
```

Aircraft number one reaches the desired heading of 55.004 degrees, so event 5 is called and heading accelerations are set to zero.

```
ss(4)=  .550042076e+02
```

```
now in event 7 at  .479543839e+02
```

Aircraft number one has reached the desired pitch of -.034 degrees, but SLAM time step tolerance allowed minor overcorrection of .004 degrees. The pitch flags are set to zero to halt further accelerations in pitch.

```
ss(5)= -.387044102e-01
```

```
now in event 3 at  .567499733e+02
```

Aircraft number one reaches its climb point. A comparison of the aircraft's current x and y coordinates is made with the coordinates of the flight planned position of the climb point. The actual lateral error is 187 feet, while the altitude error is 0.4 feet. The global xx variables are updated to give navigational data to the next point, the level-off point. Flight planned heading and pitch into the level-off point were 055 and five degrees respectively. The xx variables here are close to these values, as they indicate a desired heading of 55.002 degrees and a pitch of 4.408 degrees. Minor corrections are required so again heading and pitch flags are turned on.

```
        ss(1)= -.522327490e+04  ss(2)= -.104575117e+05
        ss(3)=  .754509811e+02  ss(4)=  .550042076e+02
        ss(5)= -.387044102e-01  Kntrl= 36
        xx(64)= -.100000000e+01  xx(65)=  1.00000000
        xx(15)= -.389966626e+04  xx(16)= -.953087598e+04
        xx(17)=  .200000000e+03  xx(18)=  .550022163e+02
        xx(19)=  4.40826082
        head1=  1.00000000  pitch1=  1.00000000
```

```
now in event 5 at  .567504730e+02
```

Aircraft number one reaches the desired heading with the SLAM tolerance, so event 5 which turns off the heading flag is called.

```
        ss(4)=  .550000496e+02
```

```
now in event 7 at  .572857437e+02
```

Aircraft number one reaches the desired pitch so event 7 called to zero out the pitch flags.

```
        ss(5)=  4.40826082
```

```
now in event 3 at  .583499870e+02
```

Aircraft number one has reached the level-off point. Lateral error from the flight planned position is 199 feet. The altitude error is 32 feet low, so the pitch flag is turned on as the aircraft climbs up toward 200 feet. Minor heading corrections are also required so the heading flags are turned on. The desired pitch of 64 degrees is so large because of the short remaining distance to the next point, the track point.

```
        ss(1)= -.406307471e+04  ss(2)= -.964520703e+04
        ss(3)=  .167989410e+03  ss(4)=  .550000496e+02
        ss(5)=  4.40826082  Kntr1= 31
        xx(64)=  1.00000000  xx(65)= -.100000000e+01
        xx(15)= -.172050977e+04  xx(16)= -.800516064e+04
        xx(17)=  .200000000e+03  xx(18)=  .550012856e+02
        xx(19)=  .641391993e+00
        head1=  1.00000000  pitch1=  1.00000000

now in event 5 at  .583504868e+02
```

Aircraft number one reaches the desired heading, so the heading flags are turned off.

```
        ss(4)=  .550042076e+02

now in event 7 at  .588038177e+02
```

Aircraft number reaches the desired pitch angle enroute to the track point. The pitch flags are turned off via event 7.

```
        ss(5)=  .641391993e+00

now in event 3 at  .613499641e+02
```

Aircraft number one reaches the track point. The lateral error is 198 feet, and the altitude error is +9.8 feet. As a result, a minor pitch correction down is programmed via the pitch flags. A minor heading correction is also scheduled.

```
        ss(1)= -.188294629e+04  ss(2)= -.811904297e+04
        ss(3)=  .209873062e+03  ss(4)=  .550042076e+02
```

```
ss(5)=  .641391993e+00  Kntr1= 26
xx(64)= -.100000000e+01  xx(65)= -.100000000e+01
xx(15)=  .190926733e+04  xx(16)= -.546380664e+04
xx(17)=  .200000000e+03  xx(18)=  .549984360e+02
xx(19)= -.122203313e+00
head1=  1.00000000  pitch1=  1.00000000
```

now in event 5 at .613506584e+02

Aircraft number one reaches the desired heading.

```
ss(4)=  .549984360e+02
```

now in event 7 at .614419632e+02

Aircraft number one reaches the desired pitch.

```
ss(5)= -.122203320e+00
```

now in event 3 at .663501358e+02

Aircraft number one reaches the first weapons release point. The lateral error is 191 feet, and altitude error is +1.2 feet. Minor heading and pitch corrections could be justified, but event 3 logic forces aircraft to maintain current heading and pitch during the release of weapons. As a result, the heading and pitch flags are correctly set to zero.

```
ss(1)=  .175110059e+04  ss(2)= -.557452832e+04
ss(3)=  .201201370e+03  ss(4)=  .549984360e+02
ss(5)= -.122203320e+00  Kntr1= 21
xx(64)=  .000000000e+00  xx(65)=  .000000000e+00
xx(15)=  .259570166e+04  xx(16)= -.498320605e+04
xx(17)=  .200000000e+03  xx(18)=  .549984360e+02
xx(19)= -.122203320e+00
head1=  .000000000e+00  pitch1=  .000000000e+00
```

now in event 3 at .673001938e+02

Aircraft number one reaches the last weapons release point. A turn of 45 degrees away from the runway center is called for in event 7. The coordinates of a future navigation point are no longer needed, so the x and y coordinates desired are arbitrarilly set to large numbers. The desired heading of 99.99 degrees correctly reflects a 45 degree right turn added to the current heading of 54.99 degrees. The pitch accelerations and pitch are zeroed as the pilot wants to avoid descent into the bombs' fragmentation pattern during the turn.

```
        ss(1)=  .244156128e+04  ss(2)= -.509107666e+04
        ss(3)=  .199501297e+03  ss(4)=  .549984360e+02
        ss(5)=  .000000000e+00  Kntrl= 16
        xx(64)=  1.00000000  xx(65)=  .000000000e+00
        xx(15)=  .120000000e+06  xx(16)=  .120000000e+06
        xx(17)=  .199501297e+03  xx(18)=  .999984360e+02
        xx(19)=  .000000000e+00
        head1=  1.00000000  pitch1=  .000000000e+00
```

now in event 9 at .727100754e+02

Aircraft number one completes the 45 degree turn as indicated by the current heading of 99.99 degrees. The desired x and y coordinates remain arbitrarily big. A descent to 50 feet altitude within 9000 feet ground range is correctly called for via event 9. In addition, position with respect to the runway center is calculated by a call to subroutine MBRAN2. The desired aircraft heading to exit the target area is set at this value of magnetic bearing. In this case, hand calculations indicate that the bears 120 degrees from the runway center. Event 9 correctly schedules 120 as the desired heading and a heading flag to the right.

```
        ss(1)=  .701140869e+04  ss(2)= -.409519678e+04
        ss(3)=  .200056046e+03  ss(4)=  .999959641e+02
        ss(5)=  .000000000e+00  Kntrl= 16
        xx(64)=  1.00000000  xx(65)= -.100000000e+01
        xx(15)=  .120000000e+06  xx(16)=  .120000000e+06
        xx(17)=  .500000000e+02  xx(18)=  .120290413e+03
        xx(19)= -.955268323e+00
        head1=  1.00000000  pitch1=  1.00000000
```

now in event 7 at .728251724e+02

Aircraft number one reaches the desired pitch to proceed to 50 feet.

```
  ss(5)= -.955268323+00
```

```
now in event 5 at  .751499863e+02
```

Aircraft number one reaches the desired heading of 120 degrees.

```
  ss(4)=  .120290413e+03
```

```
now in event 4 at  .757500153e+02
```

Aircraft number two reaches the midcourse correction. Minor heading and pitch changes are correctly scheduled.

```
  ss(6)= -.122525361e+05  ss(7)= -.163271973e+05
  ss(8)=  .797706985e+02  ss(9)=  .650000000e+02
  ss(10)=  .000000000e+00  Kntr2= 41
  xx(66)=  1.00000000  xx(67)= -.100000000e+01
  xx(25)= -.502733398e+04  xx(26)= -.129589004e+05
  xx(27)=  .750000000e+02  xx(28)=  .650038300e+02
  xx(29)= -.342911668e-01
  head2=  1.00000000  pitch2=  1.00000000
```

```
now in event 6 at  .757505188e+02
```

Aircraft number two reaches the desired heading.

```
  ss(9)=  .650041580e+02
```

```
now in event 8 at  .757541504e+02
```

Aircraft number reaches the desired pitch.

```
  ss(10)= -.342911668e-01
```

```
now in event 11 at  .830000305e+02
```

Aircraft number one reaches 50 feet altitude, so an abrupt level off is scheduled and pitch flags are set to zero.

```
  ss(3)=  .494371796e+02
```

```
now in event 4 at  .845500641e+02
```

Aircraft number two reaches the pullup point. The lateral error from the planned position is 164 feet, and the altitude error is +1 foot. The desired pitch of 14.2 degrees is close to the flight planned climb angle of 15 degrees. The pitch flag is correctly set and a minor heading correction is scheduled. The flight planned rollin altitude of 845 feet correctly becomes the desired altitude.

```
ss(6)= -.517631494e+04  ss(7)= -.130284102e+05
ss(8)=  .760006790e+02  ss(9)=  .650041580e+02
ss(10)= -.342911668e-01  Kntr2= 36
xx(66)= -.100000000e+01  xx(67)=  1.00000000
xx(25)= -.242134131e+04  xx(26)= -.117438076e+05
xx(27)=  .845412598e+03  xx(28)=  .649992294e+02
xx(29)=  .142052412e+02
head2=  1.00000000  pitch2=  1.00000000
```

now in event 6 at .845506592e+02

Aircraft number two reaches the desired heading.

```
ss(9)=  .649992294e+02
```

now in event 8 at .862657700e+02

Aircraft number two reaches the desired pitch angle.

```
ss(10)=  .142052412e+02
```

now in event 4 at .878500824e+02

Aircraft number two reaches the rollin point. The lateral error is 171 feet, and the altitude error is -232 feet. The desired altitude correctly becomes the track point altitude of 1514 feet. Event 4 forces the desired heading after the rollin point on a pop-to-angular delivery to be the flight planned release heading. This is correctly reflected by xx(28)=035 degrees.

```
ss(6)= -.257650098e+04  ss(7)= -.118161602e+05
ss(8)=  .613897949e+03  ss(9)=  .649992294e+02
ss(10)=  .142052412e+02  Kntr2= 31
xx(66)= -.100000000e+01  xx(67)=  1.00000000
xx(25)= -.915771484e+00  xx(26)= -.971304102e+04
```

```
          xx(27)=  .151479163e+04  xx(28)=  .350000000e+02
          xx(29)=  .151604347e+02
          head2=  1.00000000  pitch2=  1.00000000
```

```
now in event 8 at  .879651718e+02
```

Aircraft number two reaches the planned pitch, so pitch flags are zeroed.

```
          ss(10)=  .151604347e+02
```

```
now in event 6 at  .914567947e+02
```

Aircraft number two reaches the planned heading, so the heading flags are zeroed.

```
          ss(9)=  .350000000e+02
```

```
now in event 4 at  .916000061e+02
```

Aircraft number two reaches the track point. Lateral error is 176 feet, and altitude error is -40 feet. The desired pitch angle of -14.3 degrees reflects a computed angle of -9.3 degrees minus an additional five degrees to compensate for the transition from a positive pitch of 15 degrees to the negative pitch desired. In addition, a minor heading correction is called for.

```
          ss(6)= -.174095444e+03  ss(7)= -.974474023e+04
          ss(8)=  .148377625e+04  ss(9)=  .350000000e+02
          ss(10)=  .151604347e+02  Kntr2= 26
          xx(66)=  1.00000000  xx(67)= -.100000000e+01
          xx(25)=  .250224829e+04  xx(26)= -.613868945e+04
          xx(27)=  .745412598e+03  xx(28)=  .365781822e+02
          xx(29)= -.143377457e+02
          head2=  1.00000000  pitch2=  1.00000000
```

```
now in event 6 at  .917897415e+02
```

Aircraft number two reaches the desired heading.

```
          ss(9)=  .365781822e+02
```

```
now in event 8 at  .951542130e+02
```

Aircraft number two reaches the desired pitch angle.

```
    ss(10)= -.143377457e+02
```

```
now in event 4 at  .965500641e+02
```

Aircraft number two arrives at the first weapons release point. A total of 30.2 seconds have elapsed since aircraft number one began releasing weapons. Therefore, the problem of sufficient time between aircraft to insure against fragmentation of the second aircraft is satisfied. The lateral error of aircraft number two at this point is 171 feet, and the altitude error is +443 feet. As with event 3, event 4 requires the aircraft heading and pitch to remain constant during the release of weapons. The heading and pitch flags, therefore, correctly are zeroed.

```
    ss(6)=  .239821924e+04  ss(7)= -.627604199e+04
    ss(8)=  .118886255e+04  ss(9)=  .365781822e+02
    ss(10)= -.143377457e+02  Kntr2= 21
    xx(66)=  .000000000e+00  xx(67)=  .000000000e+00
    xx(25)=  .297533667e+04  xx(26)= -.546315088e+04
    xx(27)=  .600000000e+03  xx(28)=  .365781822e+02
    xx(29)= -.143377457e+02
    head2=  .000000000e+00  pitch2=  .000000000e+00
```

```
now in event 4 at  .975001221e+02
```

Aircraft number two reaches the final weapon release point. The lateral error is 181 feet, and the altitude error is 380 feet. The post-release turn logic is similar to that described for aircraft number one. In this case, a 45 degree right turn plus the current heading of 36.57 degrees results in a desired heading of 81.57 degrees. The heading flag is set to indicate this, and the pitch flags and the pitch are set to zero to indicate the pilot's attempt to reduce his descent during his turn away from the target.

```
ss(6)=  .288492725e+04  ss(7)= -.562026367e+04
ss(8)=  .980240845e+03  ss(9)=  .365781822e+02
ss(10)=  .000000000e+00  Kntr2= 16
xx(66)=  1.00000000  xx(67)=  .000000000e+00
xx(25)=  .120000000e+06  xx(26)=  .120000000e+06
xx(27)=  .980240845e+03  xx(28)=  .815781860e+02
xx(29)=  .000000000e+00
head2=  1.00000000  pitch2=  .000000000e+00
```

now in event 18 at .102153061e+03

Aircraft number two completes the postrelease turn. The logic is similar to that described for aircraft number one. In this case, the hand calculations indicated that a right turn to a heading of 115 degrees is required. This is correctly scheduled in event 18. In addition, a negative pitch to move the aircraft below 50 feet is correctly commanded.

```
ss(6)=  .690606885e+04   ss(7)= -.32312859e+04
ss(8)=  .980795593e+03   ss(9)=  .81575561e+02
ss(10)=  .000000000e+00  Kntr2= 16
xx(66)=  1.00000000  xx(67)= -.100000000e+01
xx(25)=  .120000000e+06  xx(26)=  .120000000e+06
xx(27)=  .500000000e+02  xx(28)=  .115076309e+03
xx(29)= -.590507126e+01
head2=  1.00000000  pitch2=  1.00000000
```

now in event 8 at .103621498e+03

Aircraft number two reaches the desired pitch.

```
ss(10)= -.590507126e+01
```

now in event 6 at .106937653e+03

Aircraft number two reaches the desired heading for exit from the area.

```
ss(9)=  .115076309e+03
```

```
now in event 12 at  .113453079e+03

     Aircraft number two abruptly levels off as it passes 50 feet

altitude.

       ss(8)=  .498600159e+02

now in event 13 at  .132403076e+03

     Aircraft number one exits the ten nautical mile radius target area.

now in event 14 at  .163350510e+03

     Aircraft number two exits the ten nautical mile radius target area.
```

APPENDIX D

THREAT LOGIC VERIFICATION

This appendix contains logic and computer code used to verify that the model's threat logic functions correctly and that each aircraft reacts properly to these threats. In this scenario, aircraft number one performs a level attack, while aircraft number two accomplishes a pop-to-angular attack. In addition, neither aircraft uses jamming, each pilot is aware of SAMs launched, and the track and acquisition times are five and eight seconds for SAM sites one and two respectively. As indicated by the following results, aircraft number one is engaged by both AAA sites and SAM site number one, while aircraft number two is engaged by both AAA sites and SAM site number two.

```
now in event 33 at  .488717117e+02
```

Aircraft number one crosses the maximum range of AAA 4. The trace results indicate the aircraft position as 10950 feet west and 14470 feet south of the runway. Hand calculations indicate this point is 9809 feet from the AAA site. The logic, therefore, has been triggered correctly. The site will fire at time 54.87 seconds.

```
  xx(41)=  .548717117e+02
```

```
now in event 20 at  .548717117e+02
```

A burst is fired by AAA 4 as scheduled at 54.87 seconds. Subroutine PKAA10 is called and the probability of kill (pK) of AAA 4 against aircraft number one is assessed.

```
  vf=  .133804761e+04  tof=  2.7693750
  pKss10=  .495730201e-03
  pKaaa=  .483761840e-01
  pK10=  .483761840e-01  acft= 1
```

```
now in event 15 at  .573857422e+02
```

Aircraft number one climbs above 100 feet. This is confirmed by the trace as the aircraft is one second beyond the climb point at an altitude of 102 feet. A check is made of the status of SAM site number one. This sites's status is correctly identified as idle, two missiles available, and confounding delay complete. The site, therefore, begins tracking the target, the site's status is changed to reflect the tracking of aircraft number one, and the tracking and acquisition time is correctly scheduled for 62.38 seconds. A similar check is made of SAM site number two. However, the tracking and acquisition completion time for site number two is a time of 65.38 seconds due to the longer track and acquisition time requirement.

```
  th( 1 ,7)=  .000000000e+00
  th( 1 ,6)=  2.00000000
  th( 1 ,8)=  .000000000e+00
  th( 1 ,7)=  1.00000000
  th( 1 ,10)=  1.00000000
  th( 1 ,11)=  5.00000000
  xx(38)=  .623857422e+02  xx(39)=  .000000000e+00
  th( 2 ,7)=  .000000000e+00
  th( 2 ,6)=  2.00000000
  th( 2 ,8)=  .000000000e+00
  th( 2 ,7)=  1.00000000
  th( 2 ,10)=  1.00000000
  th( 2 ,11)=  8.00000000
  xx(38)=  .623857422e+02  xx(39)=  .653857422e+02

now in event 17 at  .623857422e+02
```

SAM site number one is the first of the two SAM sites ready to engage aircraft number one. This ready-to-fire time corresponds to the time scheduled for completion of track and acquisition. Calculations are made to determine the current planned impact point's coordinates. Since the impact point is within the maximum missile and outside the minimum missile range, the site's first missile is fired. The scheduled should occur in one second.

```
  mslx8=.120000000e+05 msly8=-.180000000e+05 mslz8=.0000000e+00
  acftx8= -.113013293e+04  acfty8= -.759193262e+04
  acftz8=  .208295639e+03  velx8=  .726800232e+03
  vely8=  .508891693e+03  velz8= -.178953385e+01
  v8=  .101900000e+04  tofmx8=  .193999996e+02
  imptx8=.583161035e+04 impty8=-.271745215e+04 imptz8=.191154373e+03
  pch8=  .664584577e+00  hdg8=  .338024872e+03
  grimp8=  .164804531e+05  sr8=  .164815605e+05
  tti8=  9.57861996
  xx(42)=  1.00000000  xx(43)=  .000000000e+00
```

now in event 31 at .630869560e+02

Aircraft number one enters the engagement ring of AAA 3. This is confirmed by trace coordinates of the aircraft's position as 620 feet west and 7235 feet south of the runway center. This is a range of 9807 feet to this AAA site. As a result, the site is correctly scheduled to fire in six seconds at time 69.08 seconds.

```
  xx(40)=  .690869598e+02
```

now in event 23 at .633857422e+02

SAM 1 is launched as scheduled. Administrative duties are performed. These include the reduction of available SAMs at the site to one and the establishment of the destruct time for SAM 1 to 82.785 seconds. This correctly reflects the current time of 63.38 seconds plus a maximum time of flight of 19.4 seconds. In addition, the planned impact point and planned pK are updated at launch. The next scheduled impact update time is set to occur in one second.

```
  launch time missle 1
  th(1,6)=  1.00000000 xx(42)=  .100000000e+04
  msl(1,2)=  1.00000000 msl(1,4)=  .827857437e+02
  imsl= 1
  mslx8=  .120000000e+05  msly8= -.180000000e+05
  mslz8=  .000000000e+00
  acftx8= -.403323090e+03  acfty8= -.708303027e+04
  acftz8=  .206506073e+03  velx8=  .726800232e+03
  vely8=  .508891693e+03  velz8= -.178953385e+01
  v8=  .101900000e+04  tofmx8=  .194000015e+02
  imptx8=.666408838e+04 impty8=-.213456250e+04 imptz8=.189104630e+03
  pch8=  .647316337e+00  hdg8=  .341416290e+03
  grimp8=  .167386992e+05  sr8=  .167397676e+05
  tti8=  9.72400856
  rm=  .510358789e+04
  jsdb= 0  js= 0  cepf=  .164172306e+02
  pKsam9=  .999998450e+00  equation form
  xx(54+ 1 )=  .643857422e+02


now in event 23 at  .673857422e+02
```

One of the scheduled impact point updates occurs. The predicted impact point has not changed significantly as the aircraft's flight vector has been relatively constant.

```
  imsl= 1
  mslx8=  .980693164e+04  msly8= -.114747588e+05
  acftz8=  .199510391e+03  velx8=  .733075317e+03
  vely8=  .499813110e+03  velz8=  .102662377e+00
  v8=  .101900000e+04  tofmx8=  .154000015e+02
  imptx8=.667139893e+04 impty8=-.220688477e+04 imptz8=.200093994e+03
  pch8=  .712571323e+00  hdg8=  .341313416e+03
  grimp8=  .978391797e+04  sr8=  .978467480e+04
  tti8=  5.68462133
  rm=  .508202100e+04
  jsdb= 0  js= 0  cepf=  .164169407e+02
  pKsam9=  .999998450e+00  equation form
  xx(54+ 1 )=  .683857422e+02


now in event 19 at  .690869598e+02
```

As scheduled, AAA 3 is fired at aircraft number one and a pK is assessed.

```
  vf=.145951831e+04 tof=2.358574851
  pKss10=  .126347513e-04
  pKaaa=  .126282836e-02
  pK10=  .126282836e-02  acft= 1


now in event 23 at  .703857422e+02
```

After several more updates for SAM 1, the time to impact becomes less than two seconds. The output correctly reflects that the pilot, since he is aware of a missile whose pK is approximately equal to 1.0, decides to break. The desired aircraft heading is correctly assessed as the reciprocal of the missile's intercept heading. The planned impact point position is effectively frozen as the time of the next update is set at an arbitrary large number.

```
  imsl= 1
  mslx8=  .798472998e+04  msly8= -.664683301e+04
  mslz8=  .149230270e+03
  acftx8=  .495918359e+04  acfty8= -.407569800e+04
  acftz8=  .199818024e+03  velx8=  .875499573e+03
  vely8=  .143920258e+03  velz8=  .102662377e+00
  v8=  .101900000e+04  tofmx8=  .124000015e+02
  imptx8=.657240625e+04 impty8=-.381050610e+04 imptz8=.200007187e+03
  pch8=  .918183982e+00  hdg8=  .333534088e+03
  grimp8=  .316850269e+04  sr8=  .316890942e+04
  tti8=  1.84263098
  rm=  .463214551e+04
  jsdb= 0  js= 0  cepf=  .164116936e+02
  pKsam9=  .999998510e+00  equation form
  ax2=  .495918359e+04  ay2= -.407569800e+04  az2=  .199818024e+03
  bx2=  .798472998e+04  by2= -.664683301e+04  bz2=  .149230270e+03
  ac1 heading=  .806641464e+02  msl heading=  .333534088e+03
  ac1 break heading=  .153534088e+03  xx(64)=  1.00000000
  aa3=  .496969986e+02
  xx(54+ 1 )=  .100000000e+04
```

```
now in event 27 at  .722501068e+02
```

Actual detonation of SAM 1 occurs at 72.25 seconds. This is .0281 seconds after the scheduled detonation time. The delay results from the problem of scheduling events from subroutine STATE which were described in Chapter III. In this case, however, the error is only one foot. As this is a detonation calculation, the cell pK solution method is used instead of the pK equation method to assess missile actual pK. The results of the first and last cells are listed for reference. The pilot's maneuver against SAM 1, a right level turn confirmed by the trace, completely defeats the missile.

```
  target a/c=  1.00000000
  rm=  .463214551e+04
  jsdb= 0  js= 0  cepf=  .164116936e+02
  mag=  .453196350e+03
  phi=  .647711639e+02  xpos9=  .193202103e+03  zpos9=  .409951080e+03
  gzx9=  .000000000e+00  gzy9=  .000000000e+00  gzz9=  .000000000e+00
  srx9=  .193202103e+03  sry9=  .000000000e+00  srz9=  .409951080e+03
  sr9=  .453196350e+03
  cellpK( 1 )=  .000000000e+00  pK9=  .000000000e+00
  gzx9=  .402178917e+02  gzy9= -.107757454e+02  gzz9=  .000000000e+00
  srx9=  .152984207e+03  sry9=  .107757454e+02  srz9=  .409951080e+03
  sr9=  .437698730e+03
  cellpK( 109 )=  .000000000e+00  pK9=  .000000000e+00
  pKsam9=  .000000000e+00
```

```
now in event 9 at  .722501068e+02
```

Since the detonation of SAM 1 has occurred, aircraft number one terminates the defensive maneuver. The most expeditious departure away from the high threat runway environment is planned. Hand calculations confirm that the current position of the aircraft from the runway is a magnetic bearing of 123 degrees. The aircraft correctly begins a turn to a heading of 123 degrees to exit the target area.

```
  ss(1)=  .658224219e+04  ss(2)= -.426359570e+04
  ss(3)=  .200009186e+03  ss(4)=  .111678207e+03
  ss(5)=  .000000000e+00  kntr1= 16
  xx(64)=  1.00000000  xx(65)= -.100000000e+01
  xx(15)=  .120000000e+06  xx(16)=  .120000000e+06
  xx(17)=  .500000000e+02  xx(18)=  .122935257e+03
  xx(19)= -.954970002e+00
  head1=  1.00000000  pitch1=  1.00000000
```

```
now in event 17 at .723857422e+02
```

The second missile from SAM site number one, SAM 2, is ready for launch at the scheduled ten seconds after the site's first missile was ready. The planned impact point is checked as it was for SAM 1. Once again, the planned impact point is within the maximum time of flight of the missile. Therefore, SAM 2 is fired and a launch is scheduled in one second.

```
  mslx8=.120000000e+05 msly8=-.180000000e+05 mslz8=.0000000e+00
  acftx8=  .669345264e+04  acfty8= -.430952490e+04
  acftz8=  .198532486e+03  velx8=  .817624023e+03
  vely8= -.344220734e+03  velz8= -.146837301e+02
  v8=  .101900000e+04  tofmx8=  .193999996e+02
  imptx8=.121148428e+05 impty8=-.659193652e+04 imptz8=.101169609e+03
  pch8=  .508111894e+00  hdg8=  .570182800e+00
  grimp8=  .114086416e+05  sr8=  .114090898e+05
  tti8=  6.63066387
  xx(42)=  .100000000e+04  xx(43)=  1.00000000


now in event 24 at  .733857422e+02
```

SAM 2 is launched. The planned impact point and intercept is updated and placed in the array MSL. The calculated planned impact point altitude is below the minimum missile engagement altitude of 100 feet, so 100 feet is correctly set as the planned impact point altitude. The number of SAMs available at SAM site number one is reduced to zero.

```
  launch time missle 2
  th(1,6)=  .000000000e+00 xx(43)=  .100000000e+04
  msl(2,2)=  1.00000000 msl(2,4)=  .927857437e+02
  imsl= 2
  mslx8=  .120000000e+05  msly8= -.180000000e+05
  mslz8=  .000000000e+00
  acftx8=  .748182959e+04  acfty8= -.471449902e+04
  acftz8=  .184264816e+03  velx8=  .759241333e+03
  vely8= -.458876099e+03  velz8= -.140691900e+02
  v8=  .101900000e+04  tofmx8=  .194000015e+02
  imptx8=.121108545e+05 impty8=-.751222461e+04 imptz8=.100000000e+03
  pch8=  .546303272e+00  hdg8=  .599006653e+00
  grimp8=  .104883613e+05  sr8=  .104888379e+05
  tti8=  6.09690857
  rm=  .319781641e+04
  jsdb= 0  js= 0  cepf=  .164026546e+02
  pKsam9=  .999998510e+00  equation form
  xx(54+ 2 )=  .743857422e+02


now in event 24 at  .753857422e+02
```

An intermediate impact point update occurs for SAM 2. The impact point's position has not changed significantly since the launch of SAM 2 due to a relatively unchanged flight vector by aircraft number one.

```
  imsl= 2
  mslx8=  .120162373e+05  msly8= -.145580762e+05
  mslz8=  .333363838e+02
  acftx8=  .897864551e+04  acfty8= -.566697266e+04
  acftz8=  .156128036e+03  velx8=  .748123840e+03
  vely8= -.476786469e+03  velz8= -.140691900e+02
  v8=  .101900000e+04  tofmx8=  .174000015e+02
  imptx8=.120060684e+05 impty8=-.759637793e+04 imptz8=.100000008e+03
  pch8=  .548674226e+00  hdg8=  .359922943e+03
  grimp8=  .696170605e+04  sr8=  .696202490e+04
  tti8=  4.04668617
  rm=  .317198315e+04
  jsdb= 0  js= 0  cepf=  .164025707e+02
  pKsam9=  .999998510e+00  equation form
  xx(54+ 2 )=  .763857422e+02


now in event 24 at  .774857407e+02
```

The time to impact for SAM 2 decreases below two seconds. Since the pilot is aware and the pK is high, the pilot executes a defensive break maneuver. The desired heading during the break is correctly computed as 179 degrees, the reciprocal of the missile's heading. The trace confirms a hard right turn is executed. In addition, the next impact point update time is set to a large value to freeze the planned impact point to its current position.

```
  imsl= 2
  imslx8=  .120115635e+05  msly8= -.109439590e+05
  mslz8=  .682803879e+02
  acftx8=  .105496113e+05  acfty8= -.666816357e+04
  acftz8=  .126584488e+03  velx8=  .748123840e+03
  vely8= -.476786469e+03  velz8= -.140691900e+02
  v8=  .101900000e+04  tofmx8=  .153000031e+02
  imptx8=.120062979e+05 impty8=-.759652393e+04 imptz8=.100000000e+03
  pch8=  .542946100e+00  hdg8=  .359916504e+03
  grimp8=  .334743921e+04  sr8=  .334758984e+04
  tti8=  1.94711924
  rm=  .317193872e+04
  jsdb= 0  js= 0  cepf=  .164025707e+02
  pKsam9=  .999998510e+00  equation form
  ax2=  .105496113e+05  ay2= -.666816357e+04  az2=  .126584488e+03
  bx2=  .120115635e+05  by2= -.109439590e+05  bz2=  .682803879e+02
  ac1 heading=  .122485641e+03  msl heading=  .359916504e+03
  ac1 break heading=  .179916504e+03  xx(64)=  1.00000000
  aa3=  .386433563e+02
  xx(54+ 2 )=  .100000000e+04


now in event 34 at  .790030594e+02
```

While aircraft number one is in a break against SAM 2, aircraft number two enters the maximum engagement range of AAA 4. The actual range of aircraft number two from the AAA site is confirmed by hand calculations to be 9804 feet. A burst from the AAA site is correctly scheduled to occur at 85.00 seconds.

```
  xx(41)=  .850030594e+02
```

```
now in event 28 at  .794530869e+02
```

The actual detonation of SAM 2 occurs. This is .020 seconds after scheduled detonation time. The source of the error are SLAM inaccuracies similar to those described for SAM 1. In the case of SAM 2, the errors result in an aircraft position error of 17 feet at detonation time. However, this error is insignificant as the nearest of the cells used in the cell pk solution method is over 500 feet from the aircraft. As confirmed by the trace, the pilot's right level turn completely defeats the missile's effects.

```
  target a/c=  1.00000000
  rm=  .317193872e+04
  jsdb= 0  js= 0  cepf=  .164025707e+02
  mag=  .503881775e+03
  phi=  .483946686e+02  xpos9=  .334598816e+03  zpos9=  .376749878e+03
  gzx9=  .000000000e+00  gzy9=  .000000000e+00  gzz9=  .000000000e+00
  srx9=  .334598816e+03  sry9=  .000000000e+00  srz9=  .376749878e+03
  sr9=  .503881775e+03
  cellpk( 1 )=  .000000000e+00  pk9=  .000000000e+00
  gzx9=  .401955299e+02  gzy9= -.107697544e+02  gzz9=  .000000000e+00
  srx9=  .294403290e+03  sry9=  .107697544e+02  srz9=  .376749878e+03
  sr9=  .478256989e+03
  cellpk( 109 )=  .000000000e+00  pk9=  .000000000e+00
  pksam9=  .000000000e+00
```

```
now in event 9 at  .794530869e+02
```

As the detonation of SAM 2 has occurred, the defensive maneuver of aircraft number one is terminated and calculations made to exit the area.

The magnetic bearing of the aircraft to the runway is correctly assessed as 124 degrees. A left turn to 124 degrees is scheduled to move the aircraft out of the target area.

```
  ss(1)=  .116727207e+05  ss(2)= -.797322754e+04
  ss(3)=  .126786507e+03  ss(4)=  .155213043e+03
  ss(5)=  .000000000e+00  Kntrl= 16
  xx(64)= -.100000000e+01  xx(65)= -.100000000e+01
  xx(15)=  .120000000e+06  xx(16)=  .120000000e+06
  xx(17)=  .500000000e+02  xx(18)=  .124338150e+03
  xx(19)= -.488862216e+00
  head1=  1.00000000  pitch1=  1.00000000
```

now in event 20 at .850030594e+02

As scheduled, a burst is fired by AAA 4 against aircraft number two. Subroutine PKAA10 is called and the AAA pK is assessed.

```
  vf=  .106384302e+04  tof=  4.04107714
  pKss10=  .146854451e-03
  pKaaa=  .145803252e-01
  pK10=  .145803252e-01  acft= 2
```

now in event 16 at .851530685e+02

Aircraft number two climbs through 100 feet. A check is first made of the status of SAM site number one. It is correctly found that the site has no missiles ready to be fired. The status of SAM site number two is then checked. The results indicate the site is idle and has two missiles ready for engagement. The ready-to-fire time for site number two is scheduled for 93.153 seconds to reflect the track and acquisition time of 8 seconds for the site.

```
  th( 1 ,7)=  .000000000e+00
  th( 1 ,6)=  .000000000e+00
  th( 1 ,8)=  .000000000e+00
  th( 1 ,7)=  .000000000e+00
  th( 1 ,10)=  1.00000000
  th( 1 ,11)=  5.00000000
  xx(38)=  .100000000e+04  xx(39)=  .100000000e+04
  th( 2 ,7)=  .000000000e+00
  th( 2 ,6)=  2.00000000
  th( 2 ,8)=  .000000000e+00
  th( 2 ,7)=  1.00000000
  th( 2 ,10)=  2.00000000
  th( 2 ,11)=  8.00000000
  xx(38)=  .100000000e+04  xx(39)=  .931530685e+02
```

now in event 18 at .931530685e+02

As scheduled, SAM site number two is ready to fire its first SAM, SAM 3, at aircraft number two. Hand calculations confirm the planned impact position is between the site's minimum and maximum ranges. SAM 3 is fired and a launch is scheduled in one second.

```
  mslx8= -.120000000e+05  msly8=  .180000000e+05  mslz8=  .000000000e+00
  acftx8=  .634950500e+03  acfty8= -.865182324e+04
  acftz8=  .168741187e+04  velx8=  .528361572e+03
  vely8=  .711901123e+03  velz8=  .352616959e+02
  v8=  .116300000e+04  tofmx8=  .370999985e+02
  imptx8=.768265430e+04 impty8=.844074219e+03 imptz8=.215776025e+04
  pch8=  4.72459936  hdg8=  .131079300e+03
  grimp8=  .261100117e+05  sr8=  .261990195e+05
  tti8=  .133387890e+02
  xx(44)=  1.00000000  xx(45)=  .000000000e+00
```

now in event 32 at .935753860e+02

Aircraft number two enters the maximum range of AAA 3. The AAA site is scheduled to fire a burst at the aircraft after completion of the track and acquistion time in six seconds.

```
  xx(40)=  .995753860e+02
```

now in event 25 at .941530685e+02

SAM 3 is launched against aircraft number two as scheduled. The number of available missiles remaining at the site is correctly reset to one. The missile destruct time is set to 131.25 seconds which correctly reflects a maximum time of flight of 37.1 seconds for this type of SAM. The planned impact point data and intercept geometry calculations are updated.

```
  launch time missle 3
  th(1,6)=  .000000000e+00 xx(42)=  .100000000e+04
  msl(3,2)=  2.00000000 msl(3,4)=  .131253067e+03
  imsl= 3
  mslx8= -.120000000e+05  msly8=  .180000000e+05
  mslz8=  .000000000e+00
  acftx8=  .116289502e+04  acfty8= -.794048340e+04
  acftz8=  .165535071e+04  velx8=  .525862122e+03
  vely8=  .708533386e+03  velz8= -.930699463e+02
  v8=  .116300000e+04  tofmx8=  .370999985e+02
  imptx8=.814123047e+04 impty8=.146194922e+00 imptz8=.420286865e+03
  pch8=  .923999071e+00  hdg8=  .129392471e+03
  grimp8=  .260610117e+05  sr8=  .260644004e+05
  tti8=  .132702761e+02
  rm=  .794646338e+04
  jsdb= 0  js= 0  cepf=  .249937801e+02
  pKsam9=  .999723434e+00  equation form
  xx(54+ 3 )=  .951530685e+02
```

now in event 25 at .971530685e+02

One of the one second impact point update calculations for SAM 3 is performed. The updated planned impact point location has not changed significantly since missile launch due to a fairly consistent aircraft flight vector.

```
  imsl= 3
  mslx8= -.747687109e+04  msly8=  .142232930e+05
  mslz8=  .434309959e+02
  acftx8=  .270713818e+04  acfty8= -.585981934e+04
  acftz8=  .105651086e+04  velx8=  .512327026e+03
  vely8=  .690296570e+03  velz8= -.219600311e+03
  v8=  .116300000e+04  tofmx8=  .340999985e+02
  imptx8=.797247461e+04 impty8=.123456055e+04 imptz8=.100000000e+03
  pch8=  .160593182e+00  hdg8=  .130057739e+03
  grimp8=  .201838906e+05  sr8=  .201839707e+05
  tti8=  .102772951e+02
  rm=  .795018652e+04
  jsdb= 0  js= 0  cepf=  .249938049e+02
  pKsam9=  .999723434e+00  equation form
  xx(54+ 3 )=  .981530685e+02
```

now in event 19 at .995753860e+02

As scheduled, a burst is fired by AAA 3 against aircraft number two. The aircraft has completed its release of weapons, so this pK from subroutine PKAA10 does not effect the aircraft's probability of arrival.

```
  vf=  .147616248e+04  tof=  2.30774474
  pKss10=  .165028487e-04
  pKaaa=  .164969999e-02
  pK10=  .164969999e-02  acft= 2
```

```
now in event 18 at .103153069e+03
```

The second missile, SAM 4, from SAM site number two is ready for launch at the scheduled ten seconds after the site's first missile was ready. As with SAM 3, the planned impact point is checked. The required time of flight is within the maximum range of the missile, so the missile is fired and a launch is scheduled in one second.

```
  mslx8=-.120000000e+05 msly8=.180000000e+05 mslz8=.0000000e+00
  acftx8=.711996143e+04  acfty8=-.320422949e+04
  acftz8=.976362427e+03  velx8= .8811754e+03
  vely8= .988559875e+02  velz8=-.31143714e+02
  v8=.1163000000e+04  tofmx=  .370999985e+02
  imptx8=.263135000e+05 impty8=-.105097461e+04 imptz8=.297997925e+03
  pch8=  .399055034e+00  hdg8=  .116440315e+03
  grimp8=  .427885938e+05  sr8=  .427896328e+05
  tti8=  .217817459e+02
  xx(44)=  .100000000e+04  xx(45)=  1.00000000
```

```
now in event 26 at  .104153069e+03
```

SAM 4 is launched one second after the firing signal was initiated. The missile destruct time is correctly set to 141.25 seconds. The planned impact point position, planned pK, and missile intercept geometry is updated at launch. The predicted impact altitude is below 100 feet, so the impact altitude is correctly set to 100 feet, the minimum SAM engagement altitude.

```
  launch time missle 4
  th(2,6)=  .000000000e+00 xx(45)=  .100000000e+04
  msl(4,2)=  2.00000000 msl(4,4)=  .141253067e+03
  imsl= 4
  mslx8= -.120000000e+05  msly8=  .180000000e+05
  mslz8=  .000000000e+00
  acftx8=  .800216211e+04  acfty8= -.317283667e+04
  acftz8=  .897782532e+03  velx8=  .882045166e+03
  vely8= -.299166565e+02  velz8= -.911805115e+02
  v8=  .116300000e+04  tofmx8=  .370999985e+02
  imptx8=.287823242e+05 impty8=-.387764453e+04 imptz8=.100000000e+03
  pch8=  .123811647e+00  hdg8=  .118213379e+03
  grimp8=  .462799023e+05  sr8=  .462800078e+05
  tti8=  .235590687e+02
  rm=  .141097598e+05
  jsdb= 0  js= 0  cepf=  .251197319e+02
  pKsam9=  .999699831e+00  equation form
  xx(54+ 4 )=  .105153069e+03


now in event 25 at  .105153069e+03

     An intermediate impact point position update occurs for SAM 3.

  imsl= 3
  mslx8=  .493455469e+04  msly8=  .460908105e+04
  mslz8=  .462824799e+03
  acftx8=  .887855371e+04  acfty8= -.326971289e+04
  acftz8=  .806607361e+03  velx8=  .868440979e+03
  vely8= -.157190689e+03  velz8= -.911805115e+02
  v8=  .116300000e+04  tofmx8=  .260999985e+02
  imptx8=.147477021e+0 impty8=-.433204785e+04 imptz8=.190385925e+03
  pch8= -.117573237e+01  hdg8=  .132340958e+03
  grimp8=  .132756035e+05  sr8=  .132783984e+05
  tti8=  6.75825787
  rm=  .106235723e+05
  jsdb= 0  js= 0  cepf=  .250246601e+02
  pKsam9=  .999717832e+00  equation form
  xx(54+ 3 )=  .106153069e+03


now in event 26 at  .108153069e+03

     An intermediate update of planned impact position occurs for SAM 4.
 imsl= 4
 mslx8= -.541215283e+04  msly8=  .137438857e+05
 mslz8=  .168835430e+02
 acftx8=  .113444580e+05  acfty8= -.420709668e+04
 acftz8=  .533081543e+03  velx8=  .799202576e+03
 vely8= -.374398102e+03  velz8= -.911805115e+02
 v8=  .116300000e+04  tofmx8=  .330999985e+02
 imptx8=.289036094e+05 impty8=-.124329365e+05 imptz8=.100000000e+03
 pch8=  .110346422e+00  hdg8=  .127339943e+03
 grimp8=  .431601367e+05  sr8=  .431602188e+05
 tti8=  .219708347e+02
 rm=  .155436357e+05
 jsdb= 0  js= 0  cepf=  .251859875e+02
 pKsam9=  .999686778e+00  equation form
 xx(54+ 4 )=  .109153069e+03
```

```
now in event 25 at  .110553062e+03
```

As the time to impact for SAM 3 decreases to less than two seconds, the pilot, aware of a high predicted pK, decides to break against the missile. The desired heading in the break is correctly calculated to be 317 degrees, the reciprocal of the missile's intercept heading. The next predicted impact point update time is set to a large number to effectively freeze the position of the impact point.

```
  imsl= 3
  mslx8=  .122618076e+05  msly8= -.304727295e+04
  mslz8=  .226949722e+03
  acftx8=  .132624238e+05  acfty8= -.510559082e+04
  acftz8=  .314261688e+03  velx8=  .799202576e+03
  vely8= -.374398102e+03  velz8= -.911805115e+02
  v8=  .116300000e+04  tofmx8=  .207000046e+02
  imptx8=.147867764e+05 impty8=-.581969580e+04 imptz8=.140349350e+03
  pch8= -.132305181e+01  hdg8=  .137677994e+03
  grimp8=  .374990625e+04  sr8=  .375090552e+04
  tti8=  1.90734112
  rm=  .109286309e+05
  jsdb= 0  js= 0  cepf=  .250300541e+02
  pksam9=  .999716818e+00  equation form
  ax2=  .132624238e+05  ay2= -.510559082e+04  az2=  .314261688e+03
  bx2=  .122618076e+05  by2= -.304727295e+04  bz2=  .226949722e+03
  ac2 heading =  .115076775e+03  msl heading=  .137677994e+03
  ac2 break heading=  .317678009e+03  xx(66)= -.100000000e+01
  aa3=  .219002029e+03
  xx(54+ 3 )=  .100000000e+04
```

```
now in event 29 at  .112500000e+03
```

SAM 3 detonates .0397 seconds after the scheduled detonation time due to SLAM inaccuracies explained for SAM 1 and SAM 2. The delay results in an aircraft position error of 35 feet. This error is not significant, however, as the aircraft is no closer that 450 feet to any of the cells used in the cell pK calculation. The pilot's left turn defeats SAM 3.

```
  target a/c=  2.00000000
  rm=  .109286309e+05
```

```
jsdb= 0  js= 0  cepf=  .250300541e+02
mag=  .519726135e+03
phi=  .284448547e+02  xpos9=  .456991730e+03  zpos9=  .247535461e+03
gzx9=  .000000000e+00  gzy9=  .000000000e+00  gzz9=  .000000000e+00
srx9=  .456991730e+03  sry9=  .000000000e+00  srz9=  .247535461e+03
sr9=  .519726135e+03
cellpk( 1 )=  .000000000e+00  pk9=  .000000000e+00
gzx9=  .613377266e+02  gzy9= -.164344692e+02  gzz9=  .000000000e+00
srx9=  .395653992e+03  sry9=  .164344692e+02  srz9=  .247535461e+03
sr9=  .466996765e+03
cellpk( 109 )=  .000000000e+00  pk9=  .000000000e+00
pksam9=  .000000000e+00
```

now in event 10 at .112500000e+03

The defensive maneuver of aircraft number two is terminated. The aircraft changed heading from 115 to 082 degrees during the break. This is correct given the 8g lateral acceleration of the aircraft during the 1.9 second defensive maneuver. The magnetic bearing of the aircraft to the airfield is correctly computed as 109 degrees, and the aircraft heading flag indicates a right turn to this heading is scheduled.

```
ss(6)=  .149481299e+05  ss(7)= -.535734814e+04
ss(8)=  .314461365e+03  ss(9)=  .826890259e+02
ss(10)=  .000000000e+00  kntr2= 16
xx(66)=  1.00000000  xx(67)= -.100000000e+01
xx(25)=  .120000000e+06  xx(26)=  .120000000e+06
xx(27)=  .500000000e+02  xx(28)=  .109718925e+03
xx(29)= -.168325293e+01
head2=  1.00000000  pitch2=  1.00000000
```

now in event 26 at .117153069e+03

One of the intermediate one second updates occurs for SAM 4. The The position of the planned impact point has changed as a result of the aircraft's defensive maneuver and postdefensive turn in the engagement with SAM 3. The aircraft is on the desired exit heading of 109 degrees, so the predicted impact point location should not change significantly through completion of the intercept by SAM 4.

```
imsl= 4
```

```
  mslx8=  .938394629e+04  msly8=  .414991064e+04
  mslz8=  .725102692e+02
  acftx8=  .189567715e+05  acfty8= -.609750146e+04
  acftz8=  .195684830e+03  velx8=  .834732544e+03
  vely8= -.299600311e+03  velz8= -.259576225e+02
  v8=  .116300000e+04  tofmx8=  .240999985e+02
  imptx8=.292835957e+05 impty8=-.980398145e+04 imptz8=.100000000e+03
  pch8=  .648095235e-01  hdg8=  .125041222e+03
  grimp8=  .243044688e+05  sr8=  .243044824e+05
  tti8=  .123714151e+02
  rm=  .151748652e+05
  jsdb= 0  js= 0  cepf=  .251670876e+02
  pKsam9=  .999690533e+00  equation form
  xx(54+ 4 )=  .118153069e+03
```

now in event 26 at .127553062e+03

The time to impact for SAM 4 decreases to less than two seconds. The pilot, aware of the hich predicted pK, decides to break. The desired heading in the break is correctly assessed to be 304 degrees. The next time of impact point update is set to a large value to freeze the point at its current position.

```
  imsl= 4
  mslx8=  .261085547e+05  msly8= -.758132715e+04
  mslz8=  .965216141e+02
  acftx8=  .276390898e+05  acfty8= -.921376172e+04
  acftz8=  .495998955e+02  velx8=  .835090027e+03
  vely8= -.299728607e+03  velz8=  .102662377e+00
  v8=  .116300000e+04  tofmx8=  .137000046e+02
  imptx8=.292871855e+05 impty8=-.980529297e+04 imptz8=.100000000e+03
  pch8=  .513769202e-01  hdg8=  .124981583e+03
  grimp8=  .387939648e+04  sr8=  .387939819e+04
  tti8=  1.97355461
  rm=  .151759961e+05
  jsdb= 0  js= 0  cepf=  .251671429e+02
  pKsam9=  .999690533e+00  equation form
  ax2=  .276390898e+05  ay2= -.921376172e+04  az2=  .495998955e+02
  bx2=  .261085547e+05  by2= -.758132715e+04  bz2=  .965216141e+02
  ac2 heading =  .109718925e+03  msl heading=  .124981583e+03
  ac2 break heading=  .304981598e+03  xx(66)= -.100000000e+01
  aa3=  .207129745e+03
  xx(54+ 4 )=  .100000000e+04
```

now in event 30 at .129550095e+03

SAM 4 detonates. The scheduled detonation time was 129.5265 seconds. SLAM inaccuracies result in a delay of .0235 seconds. This delay results in an aircraft position error of 20 feet. The error is not significant, however, as the aircraft is greater than 450 feet from any cells used in the cell pK solution method.

```
  target a/c=  2.00000000
  rm=  .151759961e+05
  jsdb= 0  js= 0  cepf=  .251671429e+02
  mag=  .520564392e+03
  phi=  .242023315e+02  xpos9=  .474815216e+03  zpos9= .213395889e+03
  gzx9=  .000000000e+00  gzy9=  .000000000e+00  gzz9=  .000000000e+00
  srx9=  .474815216e+03  sry9=  .000000000e+00  srz9=  .213395889e+03
  sr9=  .520564392e+03
  cellpk( 1 )=  .000000000e+00  pK9=  .000000000e+00
  gzx9=  .616736717e+02  gzy9= -.165244808e+02  gzz9=  .000000000e+00
  srx9=  .413141541e+03  sry9=  .165244808e+02  srz9=  .213395889e+03
  sr9=  .465292175e+03
  cellpk( 109 )   .000000000e+00  pK9=  .000000000e+00
  pKsam9=  .000000000e+00
```

```
now in event 10 at  .129550095e+03
```

The defensive maneuver of aircraft number two effectively defeats the effects of SAM 4 and is terminated. The aircraft position is correctly calculated as a magnetic bearing of 107 degrees from the runway. The aircraft is directed to turn to this heading for exit from the area.

```
  ss(6)=  .293841426e+05  ss(7)= -.929630664e+04
  ss(8)=  .498049622e+02  ss(9)=  .764978790e+02
  ss(10)=  .000000000e+00  Kntr2= 16
  xx(66)=  1.00000000  xx(67)=  1.00000000
  xx(25)=  .120000000e+06  xx(26)=  .120000000e+06
  xx(27)=  .500000000e+02  xx(28)=  .107557182e+03
  xx(29)=  .124174089e-02
  head2=  1.00000000  pitch2=  1.00000000
```

```
now in event 14 at  .163750290e+03
```

The survival and damage probabilities for each aircraft and for the overall mission are summarized. Hand calculations indicated that the probability of damage and the probability of aircraft survival are correctly calculated based on the logic described in chapter III. Further, the overall mission damage and survivability calculations described in chapter III are correct.

```
 acft( 1 54 )=  .000000000e+00
 acft( 1 55 )=  .000000000e+00
 acft( 1 56 )=  .000000000e+00
 acft( 1 57 )=  .000000000e+00
 acft( 1 58 )=  .000000000e+00
 acft( 1 59 )=  .126282836e-02
 acft( 1 60 )=  .483761840e-01
 acft( 1 61 )=  .951623797e+00
 acft( 1 62 )=  .950422049e+00
 acft( 1 63 )=  .526000023e+00
 acft( 1 64 )=  .256383792e-01
 acft( 1 65 )=  .128333969e-01
 acft( 2 54 )=  .000000000e+00
 acft( 2 55 )=  .000000000e+00
 acft( 2 56 )=  .000000000e+00
 acft( 2 57 )=  .000000000e+00
 acft( 2 58 )=  .000000000e+00
 acft( 2 59 )=  .164969999e-02
 acft( 2 60 )=  .145803252e-01
 acft( 2 61 )=  .985419691e+00
 acft( 2 62 )=  .983794093e+00
 acft( 2 63 )=  .997362375e+00
 acft( 2 64 )=  .518694520e-01
 acft( 2 65 )=  .509783626e-01

 mission probability of acft survival=  .935019612e+00
 mission probability of target damage=  .631575584e-01
```

APPENDIX E

MAIN EFFECTS

Effect of Jamming

Level	1	2
Survival	.914	.236
Damage	.047	.047

Effect of Awareness

Level	1	2
Survival	.702	.449
Damage	.047	.047

Effect of Track & Aq

Level	1	2
Survival	.458	.692
Damage	.047	.047

Effect of Tgt Detect

Level	1	2
Survival	.575	.575
Damage	.063	.032

APPENDIX F

FACTOR INTERACTIONS

Effect of Jamming by Awareness

Level	Treatment	1	2
Survival	Awareness -1	.931	.472
	Awareness -2	.898	.000
Damage	Awareness -1	.047	.047
	Awareness -2	.047	.047

Effect of Jamming by Track & Aq

Level	Treatment	1	2
Survival	Track & Aq-1	.912	.005
	Track & Aq-2	.917	.468
Damage	Track & Aq-1	.047	.047
	Track & Aq-2	.047	.047

Effect of Awareness by Track & Aq

Level	Treatment	1	2
Survival	Track & Aq-1	.468	.449
	Track & Aq-2	.936	.449
Damage	Track & Aq-1	.047	.047
	Track & Aq-2	.047	.047

VITA

Michael J. Foley was born on 17 March 1951 in Chicago, Illinois. In 1969 he graduated from Brother Rice High School in Chicago and then attended the University of Detroit, Detroit, Michigan. In 1973 he graduated with a Bachelor of Science degree in Mathematics and received a commission through AFROTC. He completed navigator training in June 1974. He then served as an F-4E Weapon Systems Officer with the 36th Tactical Fighter Squadron at Osan AB, Korea and the 32nd Tactical Fighter Squadron at Soesterberg AB, The Netherlands. In 1978 he transitioned to the F-111 and served as an F-111F Weapons Systems Officer with the 48th Tactical Fighter Wing, RAF Lakenheath, England until entering the School of Engineering, Air Force Institute of Technology in August, 1982. Following graduation he will be assigned to Headquarters, Tactical Air Command, Langley AFB, Virginia.

Permanent address: 276 Ackerman Avenue
Emerson, New Jersey 07630

VITA

Captain Stephen G. Gress, Jr. was born on 17 January 1953 in Pittsburgh, Pennsylvania. He graduated from Boyle High School, Homestead, Pennsylvania, in 1970 and attended the United States Air Force Academy. He graduated in June 1974 with a Bachelor of Science degree in Engineering Management and received his commission. He attended pilot training at Webb Air Force Base, Texas, and became a T-38 Instructor Pilot. In 1977 Webb Air Force Base closed and he was reassigned to the United States Air Force Academy as a T-41 instructor pilot. In 1979 he was assigned to the 27th Tactical Fighter Squadron, Langley Air Force Base, Virginia, as an F-15 pilot. Following graduation from AFIT he will be stationed at Headquarters USAF, Washington, D.C. He is a senior pilot with 1800 hours of flying time. He is married to the former Marietta DiTirro of Arvada, Colorado, and has two children, Stephen, 6 years, and Angela, 4 years.

Permanent Address: 185 Roberta Drive
Munhall, Pennsylvania 15120